Computer Shop

Alfred Görgens

ATARI Player-Missile-Grafik



Springer Basel AG

Computer Shop
Band 28

Alfred Görgens

ATARI Player-Missile-Grafik

Springer Basel AG

CIP-Kurztitelaufnahme der Deutschen Bibliothek

Görgens, Alfred:
ATARI-Player-Missile-Grafik / Alfred Görgens.

(Computer-Shop ; Bd. 28)
ISBN 978-3-7643-1683-9 ISBN 978-3-0348-6611-8 (eBook)
DOI 10.1007/978-3-0348-6611-8

NE: GT

Ursprünglich erschienen bei Birkhäuser Verlag, Basel 1985

Umschlaggestaltung: Bruckmann & Partner, Basel

ISBN 978-3-7643-1683-9

Inhalt

Der Clan der Eingeweihten 7

1. Kapitel - Einführung in die Player-Missile-Grafik 9

1 Register

1.1 Allgemeine PM-Register 11
1.1.1 Bildschirmdarstellung 12
1.1.2 Basisadresse PM-Tabelle 13
1.1.3 PM-Kontrollregister 15
1.1.4 Breite der Player 15
1.1.5 Breite der Missiles 16

1.2 Positionsregister 17
1.2.1 Horizontale Position der Player 17
1.2.2 Horizontale Position der Missiles 18
1.2.3 Vertikale Position der Player und Missiles 18

1.3 Die Farbregister 19
1.4 Die Kollisionsregister 20
1.5 Das Prioritätsregister 22

1.6 Player-Missile-Demonstrationen 24
1.6.1 Der erste Entwurf 24
1.6.2 Bewegung 27
1.6.3 Kollision 33
1.6.4 Zusammengefügte Player 34
1.6.5 Fünfter Player aus vier Missiles 37

2. Kapitel - Anwendungen 39

2.1 Player als "Maus" 41
2.2 Player-Missile-Grafik als Spielelemente 47

2.3 Nützliche Bildschirmroutinen 52
2.3.1 Page-Flipping 52
2.3.2 Laderoutine 55
2.3.3 Vertikales Scrollen 57
2.3.4 Horizontales Scrollen 59

2.4 Player als Cursor 60
2.5 Player als Textzeichen 69

3. Kapitel - Utilities 77

3.1 Player-Missile-Editor 79
3.2 Testbild 86

Anhang 91

Player-Missile-Entwurfsraster 91
Tastatur-Code-Tabelle 92/93
ASCII-Code-Tabelle 94/95

Der Clan der Eingeweihten

Nun gibt es schon seit Jahren ATARI-Computer. Ebenso gibt es seit Jahren verblüffende Programme mit hinreißender Grafik. Wer im treuen Glauben an die Bedienungsanleitung die Möglichkeiten des ATARI zu nutzen suchte, konnte bislang nur mit staunenden Augen die Leistungen fremder Software bestaunen.

"Das schaffe ich nie", werden Sie gesagt haben. Irrtum. Hochauflösende Grafik, Bilder, die wie im Spielfilm vor den Augen vorbeirollen, oder das Wechseln von Bildschirmseiten mit einem "flash" - alles ist programmierbar. Sogar in BASIC. Mit diesem Buch wird Ihnen der Leitfaden zum vollen Ausschöpfen der ATARI-Leistungen in die Hände gegeben. (Da nimmt aber einer den Mund voll.) Stimmt nicht. Denn Player-Missile-Grafik und die weiterentwickelten Grafikfunktionen wie endloses Bildrollen oder blitzschneller Bildschirmwechsel wurden bislang noch nie so konsequent behandelt wie in diesem Buch. Man konnte den Eindruck gewinnen, daß die befähigten Programmierer die Geheimnisse ihrer Supersoftware hüten wie den Gral. (Ist doch klar: Hardmoney for Software.) Na schön. Aber inzwischen hat sich herumgesprochen, daß die Computerbesitzer selbst Programme schreiben wollen; und zwar solche, die über das Niveau von PRINT "ICH BIN DEIN COMPUTER" hinausgehen.

Gefragt sind Anwenderprogramme, Utilities und hochkarätige Spiele. In den meisten Programmen läßt sich Player-Missile-Grafik vorteilhaft verwenden. Man kann sie z.B. als Cursor programmieren, der unabhängig vom übrigen Bildschirminhalt arbeitet. Dadurch entfällt das Löschen und Wiederauffüllen von Bildschirmpunkten, wie es bei einem gePRINTeten Cursor notwendig ist. Für Spiele lassen sich feinstrukturierte Figuren (Player) und Geschosse (Missilies) definieren und in jeweils verschiedenen Farben und Größen (normal-, zweifach- und vierfach breit) auf dem Bildschirm darstellen. Und da sich Player und Missiles über die gesamte Höhe des Bildschirms erstrecken können, ist auch ein Einsatz für verschiedenfarbige Balkendiagramme möglich.

(Und das soll alles in diesem Buch stehen?) Na klar. Im ersten

Abschnitt finden Sie eine allgemeine Einführung in die Programmierung von Player-Missile-Grafik mit ersten Entwürfen, Bewegungen und der gesamten Kollisionsproblematik. Der zweite Abschnitt zeigt dann praktische Anwendungen: Player im Einsatz als "Maus", blitzschneller Bildschirmwechsel (Page-Flipping) und ein Beispiel für ein Computerspiel mit Supergrafik. Im dritten Abschnitt finden Sie schließlich Utility-Programme, die zum einen die Programmierung von Player-Missile-Grafik erleichtern und zum anderen ein bequemes Auswählen der Farbkombinationen für ein Programm ermöglichen. Die ATARI-Computer stellen immerhin 256 Farben zur Verfügung. (Da kann man ja wirklich neugierig werden.) Worauf warten Sie noch? Blättern Sie schnell weiter, bevor es ein anderer liest!

1. Kapitel

Einführung in die Player-Missile-Grafik

1 Die Register

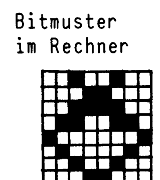
Wie programmiert man Player-Missile-Grafik? - Da stellen wir uns erst einmal ganz dumm und sagen: Player-Missile-Grafik programmiert man durch Umsetzung aller relevanten Daten zur hardwaremäßigen Prozeßsteuerung. Hierzu zählt die Reservierung von Speicherplatz für die sog. Player-Missile-Tabelle, Festlegung der Form der Player, Aktivierung der DMA (Direct Memory Access), Bestimmung der Farbe und Breite von Playern und Missiles, Abfragen der Kollisionsregister und Festlegung der Prioritäten zwischen den einzelnen Playern, Missiles und den Hintergrundfarben.

Und für alles gibt es Register. (Hätte man sich schon denken können.) Da dieser erste Abschnitt sich mit diesen Registern beschäftigt, mag Ihnen das Thema Player-Missile-Grafik zunächst wie Trockenfutter vorkommen. Es ist jedoch sinnvoller, die Theorie der komplexen Materie "in einem Rutsch" abzuhandeln, damit Sie bei Ihren eigenen Arbeiten rasch die notwendigen Informationen wiederfinden.

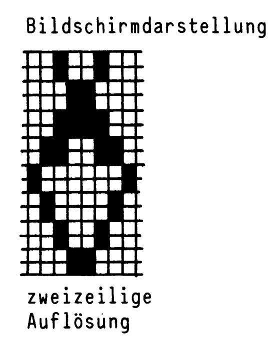
1.1 Allgemeine PM-Register

Zur Darstellung der Player-Missile-Grafik stehen zwei Auflösungen zur Verfügung. Dies bezieht sich jedoch nur auf die Höhe. Bei der zweizeiligen Darstellung beansprucht der Rechner 1024 Byte für die gesamte Player-Missile-Tabelle (vier Player, vier Missiles); bei einzeiliger Darstellung 2048 Byte. Jeder Player und jedes Missile kann insgesamt 256 Bildschirmzeilen hoch sein. Bei einzeiliger Auflösung wird jede einzelne Zeile in der PM-Tabelle definiert und auf dem Bildschirm dargestellt; bei zweizeiliger Auflösung werden nur 128 Zeilen (Byte) definiert und auf dem Bildschirm doppelt hoch dargestellt. Folgende Skizze verdeutlicht, was gemeint ist:

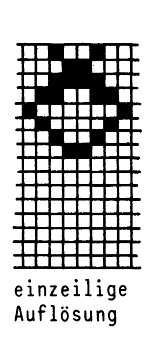
Bitmuster
im Rechner



Bildschirmdarstellung



zweizeilige
Auflösung



einzeilige
Auflösung

1.1.1 Bildschirmdarstellung (559)

Über die Adresse 559 wird der direkte Speicherzugriff auf den sog. ANTIC-Chip gesteuert. Dieser Chip verwaltet die gesamte Bildschirmdarstellung sowohl für die Grafikbetriebsarten 0 bis 15 als auch für die Player-Missile-Grafik. Der Normalwert (beim Einschalten) ist 34. Durch POKE 559,0 können Sie den Bildschirm abschalten. Der Rechner arbeitet dann ca. 30% schneller. Das ist sinnvoll, wenn Sie z.B. größere Mengen Daten einlesen oder umfangreiche Rechenoperationen durchführen, zu denen keine Bildschirmdarstellung erforderlich ist.

Die Adresse 559 hat für die Player-Missile-Grafik eine Basisbedeutung. Hier wird bestimmt, ob und mit welcher Auflösung die Player und Missiles dargestellt werden. Wenn in 559 der Ursprungswert 34 vorliegt, bleiben sämtliche Angaben in den übrigen PM-Registern wirkungslos.

Speicherstelle 559		
Bit	dez	Bedeutung
7	128	-
6	64	-
5	32	DMA Befehlsaktivierung
4	16	Einzeilige PM-Grafik
3	8	Player-Darstellung
2	4	Missile-Darstellung
1	2	Normales Bildschirmfenster
0	1	Schmales Bildschirmfenster

Befehlscodex Speicherstelle 559		
Bit	dez	Darstellung
5,1, 2	38	Missiles, zweizeilige Auflösung
5,1, 2,4	54	Missiles, einzeilige Auflösung

Fortsetzung nächste Seite

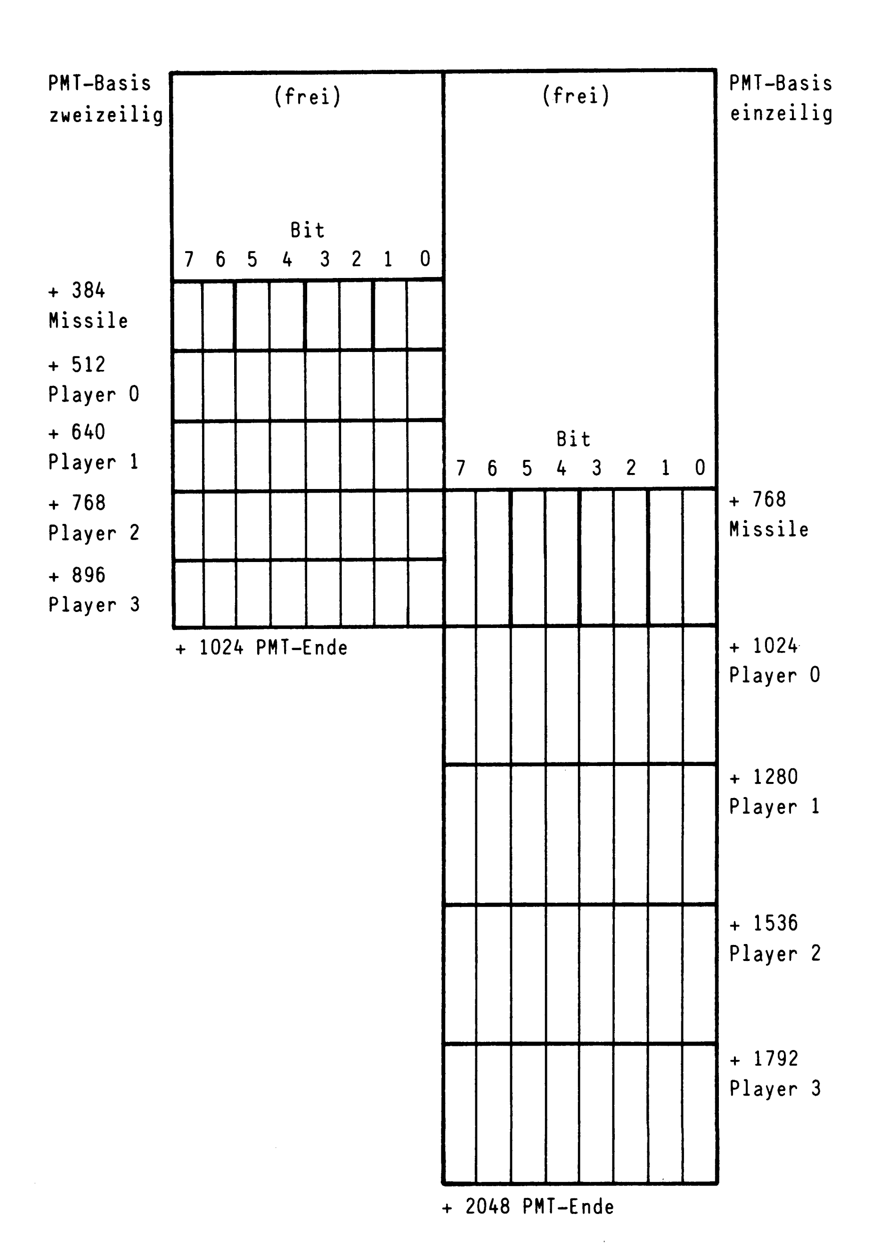
5,1, 3	42	Player, zweizeilige Auflösung
5,1, 3,4	58	Player, einzeilige Auflösung
5,1,2, 3	46	Player und Missiles, zweizeilige Auflösung
5,1,2, 3,4	62	Player und Missiles, einzeilige Auflösung
5,0,1	35	Breites Bildschirmfenster
5,1	34	Normales Bildschirmfenster
5,0	33	Schmales Bildschirmfenster

1.1.2 Basisadresse PM-Tabelle (54279)

Die Daten für die Form der Player und Missiles müssen in einem sicheren RAM-Bereich gespeichert werden. Hierzu wird je nach Auflösung 1K-Byte oder 2K-Byte vom RAMTOP (Adresse 106) aus reserviert; also von der obersten Adresse des verfügbaren RAM-Bereichs. Da die Speicherstelle 106 jeweils das High-Byte des RAMTOP enthält, wird der betreffende Dezimalwert mit 256 multipliziert. Soll nun 1K-Byte reserviert werden (zweizeilige Auflösung), lautet die Anweisung POKE 106,PEEK(106)-4. Man spricht hierbei auch von "Seiten" (4 Seiten a 256 Byte = 1024 Byte). Bei einzeiliger Auflösung lautet die Anweisung POKE 106,PEEK(106)-8.

Der Dezimalwert, der sich nun in 106 befindet, muß als Startadresse in 54279 gePOKEt werden; also POKE 54279,PEEK(106). Auch bei der Darstellung von nur einem Player muß eine gesamte PM-Tabelle reserviert werden.

Auf der folgenden Seite finden Sie eine grafische Darstellung der Player-Missile-Tabelle für zwei- und einzeilige Auflösung. Die Missiles nehmen jeweils zwei Bit ein. Jedes Byte ist dadurch aufgeteilt in Missile 3, Missile 2, Missile 1 und Missile 0.


PMT-Basis
zweizeilig
(frei)
(frei)
PMT-Basis
einzeilig
Bit
7 6 5 4 3 2 1 0
+ 384
Missile
+ 512
Player 0
+ 640
Player 1
+ 768
Player 2
+ 896
Player 3
+ 1024 PMT-Ende
Bit
7 6 5 4 3 2 1 0
+ 768
Missile
+ 1024
Player 0
+ 1280
Player 1
+ 1536
Player 2
+ 1792
Player 3
+ 2048 PMT-Ende

1.1.3 PM-Kontrollregister (53277)

Zusätzlich zum Register 559, mit dem die Player-Missile-Darstellung gewissermaßen "grundsätzlich" ermöglicht wird, können über das Grafikkontrollregister 53277 im Programmablauf Player und Missiles sichtbar werden oder mit einem Schlag verschwinden. Wenn Sie hier stattdessen versuchten sollten, den Normalwert 34 in 559 einzuPOKEn, um die PM-Grafik auszuschalten, wird es zu flimmernder Streifenbildung auf dem Bildschirm kommen. Es ist daher besser, immer nur das PM-Kontrollregister anzusprechen.

Speicherstelle 53277		
Bit	dez	Darstellung
1	2	Player
0	1	Missiles
1,0	3	Player und Missiles

1.1.4 Breite der Player (53256/57/58/59)

Neben der normalen Breite von acht Bildschirmspalten (entspr. GRAPHICS 8) können Sie jeden Player auch in doppelter oder vierfacher Breite darstellen lassen. Ähnlich wie in der Zweizeilen-Auflösung nehmen dabei die einzelnen Bits zwei oder vier Spalten des Bildschirms ein. Die Register 53256/57/58/59 sind dabei den Playern 0 bis 3 zuzuordnen.

Speicherstelle 53256/57/58/59		
Bit	dez	Bedeutung
1	2	normale Breite
0	1	doppelte Breite

Fortsetzung nächste Seite

Befehlscodex Speicherstelle 53256/57/58/59		
Bit	dez	Darstellung
	0	normale Breite
1	2	normale Breite
0	1	doppelte Breite
1,0	3	vierfache Breite

1.1.5 Breite der Missiles (53260)

Wie Sie schon aus der Player-Missile-Tabelle ersehen können, ist jedes Missile nur zwei Bit breit. Sie haben jedoch auch hier die Möglichkeit, doppelte oder vierfache Breite einzuschalten.

Speicherstelle 53260		
Bit	dez	Bedeutung
7	128	Missile 3 normal breit
6	64	Missile 3 doppelt breit
5	32	Missile 2 normal breit
4	16	Missile 2 doppelt breit
3	8	Missile 1 normal breit
2	4	Missile 1 doppelt breit
1	2	Missile 0 normal breit
0	1	Missile 0 doppelt breit

Befehlscodex Speicherstelle 53260		
Bit	dez	Bedeutung
alle	255	alle Missiles vierfach breit
7,6	192	Missile 3 vierfach breit

Fortsetzung nächste Seite

5,4	48	Missile 2 vierfach breit
3,2	12	Missile 1 vierfach breit
1,0	3	Missile 0 vierfach breit
6,4, 2,0	85	alle Missiles doppelt breit

1.2 Positionsregister

Anders als das Männlein im Walde stehen die Player nicht still und stumm auf dem Bildschirm herum. Sie lassen sich in alle Richtungen bewegen, wobei horizontal und vertikal 256 Positionen eingenommen werden können. Allerdings liegen je nach Fernsehgerät bzw. Monitor nur die Werte von ca. 40 bis 210 (horizontal) und ca. 10 bis 115 (vertikal) im sichtbaren Bildschirmbereich. Wenn Sie einen Player z.B. in die Horizontalposition 0 "schicken", verschwindet er vom Bildschirm.

1.2.1 Horizontale Position der Player

Register	Zuordnung
53248	Player 0
53249	Player 1
53250	Player 2
53251	Player 3

1.2.2 Horizontale Position der Missiles

Register	Zuordnung
53252 53253 53254 53255	Missile 0 Missile 1 Missile 2 Missile 3

1.2.3 Vertikalposition (53276)

Im Gegensatz zur horizontalen Positionsveränderung, bei der einfach der gewünschte Dezimalwert in das betreffende Register gePOKEt werden muß, lassen sich die Player und Missiles vertikal nur verschieben, indem das Bitmuster des entsprechenden Players oder Missiles neu eingelesen wird. Näheres erfahren Sie im Abschnitt 1.6.2 (Bewegung).

Ein anderes Problem ergibt sich bei Anwendung der Zweizeilen-Auflösung: Da auch die vertikale Bewegung zweizeilenweise vollzogen wird, entstehen ungleichmäßige Bewegungsabläufe. Mit Hilfe des Registers 53276 können die PM-Objekte jedoch wieder um eine Bildschirmzeile nach oben oder unten rückversetzt werden.

Speicherstelle 53276		
Bit	dez	Zuordnung
7 6 5 4 3 2 1 0	128 64 32 16 8 4 2 1	Player 3 Player 2 Player 1 Player 0 Missile 3 Missile 2 Missile 1 Missile 0

1.3 Die Farbregister (704/5/6/7)

Zur Bestimmung der Farbe stehen vier Register zur Verfügung (Player+Missile 0 bis 3). Die Farbe eines Players und des entsprechenden Missiles ist dabei identisch. Genau wie in den anderen Grafikbetriebsarten lassen sich 16 Farben in je 8 verschiedenen Helligkeitsstufen bestimmen.

Speicherstelle 704/5/6/7		
Bit	dez	Bedeutung
7	128	Blau II
6	64	Rosa
5	32	Orange
4	16	Gold
3,2,1	2-14	Helligkeiten
0	1	Helligkeit (nur GRAPHICS 9)

Befehlscodex Speicherstelle 704/5/6/7		
Bit	dez	Darstellung
7,6, 5,4	240	Hellorange
7,6,5	224	Grünorange
7,6,4	208	Gelbgrün
7,6	192	Grün
7,5,4	176	Blaugrün
7,5	160	Türkis
7,4	144	Hellblau
7	128	Blau II
6,5,4	112	Blau I

Fortsetzung nächste Seite

6,5	96	Purpur
6,4	80	Violett
6	64	Rosa
5,4	48	Rot
5	32	Orange
4	16	Gold
-	0	Grau/Schwarz

1.4 Die Kollisionsregister

Durch die gleichzeitige Darstellung mehrerer Player und Missiles und die möglichen Bewegungen der einzelnen Objekte kann es zu Berührungen kommen. Oft ist es sinnvoll, diese Kollisionen zu registrieren (z.B. in Spielprogrammen) und in entsprechende Unterprogramme zu springen.

Es gibt vier Arten von Kollisionen: 1) Missile/ Hintergrund, 2) Player/Hintergrund, 3) Missile/Player, 4) Player/Player. Für jede dieser Kollisionsarten gibt es wiederum vier Möglichkeiten; z.B. Missile 0/Hintergrund 0 (entspr. SETCOLOR 0), Missile 0/Hintergrund 1 (entspr. SETCOLOR 1), Missile 1/Player 0, Missile 1/Player 1 usw.

Insgesamt werden daher sechzehn Kollisionsregister benötigt, in denen jeweils die vier niederwertigen Bits die verschiedenen Kollisionen registrieren (Adressen 53248 bis 53263). Da es sich dabei um die gleichen Register handelt, in denen auch die horizontalen Positionen der Player und Missiles sowie die Breite der Objekte festgelegt werden, dürfen die Kollisionen nur gePEEKt und nicht etwa durch POKE simuliert werden.

Register	Kollisionsart
53248	Missile 0/Hintergrund

Fortsetzung nächste Seite

53249 53250 53251	Missile 1/Hintergrund Missile 2/Hintergrund Missile 3/Hintergrund
53252 53253 53254 53255	Player 0/Hintergrund Player 1/Hintergrund Player 2/Hintergrund Player 3/Hintergrund
53256 53257 53258 53259	Missile 0/Player Missile 1/Player Missile 2/Player Missile 3/Player
53260 53261 53262 53263	Player 0/Player Player 1/Player Player 2/Player Player 3/Player

Bitschaltung der Kollisionsregister (Hintergrund)		
Bit	dez	Kollisionsart (Objekt mit...)
3	8	SETCOLOR 3
2	4	SETCOLOR 2
1	2	SETCOLOR 1
0	1	SETCOLOR 0
-	0	keine Kollision

Bitschaltung der Kollisionsregister (Player)		
Bit	dez	Kollisionsart (Objekt mit...)
3	8	Player 3
2	4	Player 2
1	2	Player 1
0	1	Player 0
-	0	keine Kollision

Besonders wichtig ist zusätzlich das Register 53278. Mit dieser Speicherstelle werden alle Kollisionsregister gelöscht:

Sobald sich zwei PM-Objekte berühren, nehmen die o.g. Register einen entsprechenden Wert an. Dieser Wert verändert sich jedoch nicht bei erneuter Kollision. Erst wenn in Speicherstelle 53278 irgendein Wert hineingePOKEt wird (z.B. 0,1 oder 255), können neue Kollisionen registriert werden.

1.5 Das Prioritätsregister (623)

Vielleicht haben Sie bei einigen Spielprogrammen schon beobachtet, daß bestimmte PM-Objekte während ihrer Bewegung über andere Bildschirmgrafiken hinweggleiten oder dahinter verschwinden. Mit dem Register 623 können Sie die Prioritäten der Player, Missiles und Farbregister des Bildschirms (SETCOLOR 0 bis 3) sehr differenziert programmieren. Es ist z.B. möglich, daß Player 0 Priorität vor Player 1 hat, während die Grafik aus Farbregister 1 (SETCOLOR 0) Priorität vor Player 2 hat. Wenn Sie mit dieser Vorgabe z.B. ein Labyrinth auf den Bildschirm zeichnen und die Player zufällig bewegen lassen, gleitet Player 2 hinter der Labyrinthwand hinweg, während Player 0 und 1 vor ihr vorübergleiten. Wenn hingegen Player 0 und 1 kollidieren, so erscheint Player 0 im Vordergrund und Player 1 dahinter.

Das bringt sehr reizvolle Effekte in ein Programm. Für die Bestimmung der Prioritäten stehen die vier niederwertigen Bits des Registers 623 zur Verfügung. Die Rangordnung gilt immer für die Player und die zugeordneten Missiles. Sie dürfen jedoch nicht alle Prioritäten gleichzeitig aktivieren, da sonst die Objekte bei Berührung lediglich schwarz erscheinen. Mit Bit 4 erreichen Sie, daß sich die vier Missiles zu einem fünften Player zusammenschließen (zuständiges Farbregister ist 712). Bit 5 bewirkt bei Kollision von Player 0 und 1 sowie von Player 2 und 3 keine Prioritätsordnung, sondern eine Überdeckung der betreffenden Farbwerte, so daß Ihnen z.B. ein zweifarbiges Männlein zur Verfügung steht.

Speicherstelle 623		
Bit	dez	Bedeutung
5	32	Farbüberdeckung Player 0 und 1 sowie Player 2 und 3

Fortsetzung nächste Seite

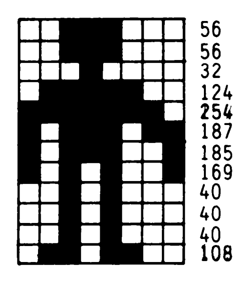
4	16	Zusammenschluß von Missiles 0,1,2,3 zu Player 4
3	8	SETCOLOR 0 vor: SETCOLOR 1,2,3 und Player 0,1,2,3 SETCOLOR 1 vor: SETCOLOR 2,3 und Player 0,1,2,3 Player 0 vor: Player 1,2,3 und SETCOLOR 2,3 Player 1 vor: Player 2,3 und SETCOLOR 2,3 Player 2 vor: Player 3 und SETCOLOR 2,3 Player 3 vor: SETCOLOR 2,3 SETCOLOR 2 vor: SETCOLOR 3
2	4	SETCOLOR 0 vor: SETCOLOR 1,2,3 und Player 0,1,2,3 SETCOLOR 1 vor: SETCOLOR 2,3 und Player 0,1,2,3 usw.
1	2	Player 0 vor: Player 1,2,3 und SETCOLOR 0,1,2,3 Player 1 vor: Player 2,3 und SETCOLOR 0,1,2,3 SETCOLOR 0 vor: SETCOLOR 1,2,3 und Player 2,3 SETCOLOR 1 vor: SETCOLOR 2,3 und Player 2,3 SETCOLOR 3 vor: Player 2,3 Player 2 vor: Player 3
0	1	Player 0 vor: Player 1,2,3 und SETCOLOR 0,1,2,3 Player 1 vor: Player 2,3 und SETCOLOR 0,1,2,3 usw.

1.6 Player-Missile Demonstrationen

Nachdem Sie nun alle Register zur Darstellung von Player-Missile-Grafik kennengelernt haben, können Sie sich an die praktische Programmierarbeit heranmachen. Nachfolgend finden Sie einige Beispiele für ein- und zweizeilige Auflösung, horizontale und vertikale Bewegung, Kollisionen und Prioritätsdemonstrationen. Sie können mit diesen Beispiel-Listings nach Herzenslust herumexperimentieren, eigene Figuren entwerfen oder die Prioritäten anders verteilen.

1.6.1 Der erste Entwurf

Jeder Player ist 8 Bit breit. Für die Höhe stehen bei zweizeiliger Auflösung 128 Linien (=128 Byte) und bei einzeiliger Auflösung 256 Linien (=256 Byte) zur Verfügung. In der Praxis werden Sie diese Höhe wohl nie nutzen; meistens sind PM-Objekte 8 bis 40 Zeilen hoch. Das auf der folgenden Seite abgebildete Männlein beansprucht z.B. 12 Zeilen.



Um diesen Entwurf als Player auf dem Bildschirm erscheinen zu lassen, ist folgendes Programm erforderlich:

```
0 REM PM-DEMO 1 - ZWEIZEILIG
10 POKE 106,PEEK(106)-4
20 GRAPHICS 5:? "MOMENT BITTE"
30 POKE 54279,PEEK(106)
40 VERTIKAL=50:HORIZONTAL=100
50 PMTABELLE=PEEK(106)*256
60 FOR LOESCHEN=PMTABELLE TO PMTABELLE+1024
70 POKE LOESCHEN,0
80 NEXT LOESCHEN
90 PLAYER0=PMTABELLE+512
100 FOR BYTE=0 TO 11
110 READ DATEN
```

```
120 POKE PLAYER0+BYTE+VERTIKAL,DATEN
130 NEXT BYTE
140 POKE 559,42
150 POKE 53277,2
160 POKE 704,255:REM HELLORANGE
170 POKE 53248,HORIZONTAL
180 ? CHR$(125)
190 GOTO 190
200 DATA 56,56,16,124,254,187,185,169,40,40,40,108
```

10: Vom RAMTOP aus werden vier "Seiten" (*256 Byte = 1024 Byte) für die Player-Missile-Tabelle reserviert.
20: Erst nach dieser Reservierung wird die gewünschte Grafikbetriebsart programmiert.
30: In Register 54279 wird der Anfang der PM-Tabelle eingegeben.
40: Die vertikale und horizontale Position für das Männlein.
50: Um die tatsächliche Anfangsadresse der PM-Tabelle zu ermitteln, wird der Wert aus Register 106 mit 256 multipliziert.
60 bis 80: Es kann vorkommen, daß sich irgendwelche Dezimalwerte in den Adressen der Tabelle befinden. Zu Programmbeginn werden daher alle Speicherstellen auf Null gesetzt.
90: Wie Sie aus der Player-Missile-Tabelle (Abschnitt 1.1.2) ersehen können, beginnen die Speicherstellen für Player 0 (in diesem Fall das Männlein) bei 512.
100 bis 130: Die zwölf Dezimalwerte für das Bitmuster des Männleins werden gelesen und in die PM-Tabelle eingetragen. Das hier gezeigte Prinzip gilt für alle Player und Missiles.
140: Der Bildschirm wird für die Darstellung der PM-Grafik vorbereitet (vergl. Abschnitt 1.1.1).
150: Ebenso wird das PM-Kontrollregister für die Bildschirmdarstellung vorbereitet (vergl. Abschnitt 1.1.3).
160: Die Farbe des Players wird festgelegt.
170: Mit dem Register 53248 läßt sich die horizontale Position von Player 0 festlegen. Sie können übrigens die Zeilen 140 bis 170 auch gleich hinter den GRAPHICS-Befehl legen. Das Programm funktioniert dann ebenfalls.

Starten Sie mit RUN (RETURN). Nach etwa zehn Sekunden erscheint die Figur auf dem Bildschirm. Unterbrechen Sie (wenn Sie sich "satt" gesehen haben) mit BREAK und geben Sie ohne Zeilennummern ein:

```
POKE 53256,1 (RETURN)
POKE 53256,3 (RETURN)
POKE 53256,0 (RETURN)
```

Sie sehen dann die Wirkung bei doppelter und vierfacher Breite (vergl. Abschnitt 1.1.4). Falls Sie das Männlein mit einzeiliger Auflösung programmieren wollen, geben Sie folgendes Programm ein:

```
0 REM PM-DEMO 2 - EINZEILIG
10 POKE 106,PEEK(106)-8
20 GRAPHICS 5:? "MOMENT BITTE"
30 POKE 54279,PEEK(106)
40 VERTIKAL=50:HORIZONTAL=100
50 PMTABELLE=PEEK(106)*256
60 FOR LOESCHEN=PMTABELLE TO PMTABELLE+2048
70 POKE LOESCHEN,0
80 NEXT LOESCHEN
90 PLAYER0=PMTABELLE+1024
100 FOR BYTE=0 TO 11
110 READ DATEN
120 POKE PLAYER0+BYTE+VERTIKAL,DATEN
130 NEXT BYTE
140 POKE 559,58
150 POKE 53277,2
160 POKE 704,255:REM HELLORANGE
170 POKE 53248,HORIZONTAL
180 ? CHR$(125)
190 GOTO 190
200 DATA 56,56,16,124,254,187,185,169,40,40,40,108
```

10: Vom RAMTOP aus müssen 8 "Seiten" (*256 Byte = 2048 Byte) reserviert werden.
60: Zum Löschen der PM-Tabelle müssen nun 2048 FOR...NEXT-Durchläufe programmiert werden.
90: Im Gegensatz zur zweizeiligen Auflösung, wo die Speicherstellen für Player 0 bei 512 beginnen, liegen sie bei einzeiliger Auflösung im Bereich von 1024 bis 1279.
140: Die Aktivierung der Bildschirmdarstellung von Playern mit einzeiliger Auflösung (vergl. Abschnitt 1.1.1).

Auch hier sollten Sie nach RUN (RETURN) mit BREAK unterbrechen und mit den Anweisungen

```
POKE 53256,1 (RETURN)
POKE 53256,3 (RETURN)
POKE 53256,2 (RETURN)
```

die Wirkung der doppelten und vierfachen Verbreiterung betrachten.

1.6.2 Bewegung

Interessant wird Player-Missile-Grafik, wenn die einzelnen Objekte durch Joystick oder innerhalb eines Programmablaufs bewegt werden. Falls Sie einen Player nur horizontal bewegen wollen, müssen Sie folgendes Prinzip anwenden:

```
0 REM PM-DEMO 3 - BEWEGUNG
10 POKE 106,PEEK(106)-4
20 GRAPHICS 5:? "MOMENT BITTE"
30 GOSUB 100
40 H=50
50 J=STICK(0)
60 IF J=7 AND H<255 THEN H=H+1
70 IF J=11 AND H>0 THEN H=H-1
80 POKE 53248,H
90 GOTO 50
100 POKE 54279,PEEK(106)
110 PMT=PEEK(106)*256
120 FOR L=PMT TO PMT+1024:POKE L,0:NEXT L
130 P0=PMT+512
140 FOR L=0 TO 11:READ D:POKE P0+L+50,D:NEXT L
150 POKE 559,42
160 POKE 53277,2
170 POKE 704,255
180 ? CHR$(125):RETURN
190 DATA 56,56,16,124,254,187,185,169,40,40,40,108
```

40: Die horizontale Anfangsposition des Männleins ist 50.
50: Die Abfrage des Joysticks (Port 1).
60: Solange der Steuerknüppel nach rechts gedrückt wird und die horizontale Position des Players den Wert 255 noch nicht überschritten hat, soll H um Eins erhöht werden.
70: Solange der Steuerknüppel nach links gedrückt wird und H größer Null ist, soll H Eins abgezogen werden.
80: Die aktuelle Horizontalposition wird gePOKEt.
100 bis 190: Vergl. 90 bis 190 im PM-DEMO 1.

Nach diesem Prinzip können Sie auch alle anderen Player entwerfen und (zunächst nur horizontal) bewegen. (Und wann, um alles in der Welt, kommen endlich die Missiles ins Spiel?) Nun, die Programmierung von Missiles unterscheidet sich kaum von der Player-Darstellung. Je nach Auflösung stehen Ihnen 128 oder 256 Byte für ein Missile zur Verfügung. Die Breite beträgt 2 Bit (vergl. PM-Tabelle im Abschnitt 1.1.2).

Im folgenden Beispielprogramm wird zum Player 0 (Männlein)

noch Missile 0 hinzugefügt, das sich durch Knopfdruck (Joystick) von dem Männlein löst und mit einem schußartigen Geräusch nach rechts aus dem Bildschirm bewegt.

```
0 REM PM-DEMO 4 - BEWEGUNG+MISSILE
10 POKE 106,PEEK(106)-4
20 GRAPHICS 5:? "MOMENT BITTE"
30 GOSUB 110
40 H=50:HM=55
50 J=STICK(0):K=STRIG(0)
60 IF J=7 AND H<250 THEN H=H+1:HM=HM+1
70 IF J=11 AND H>5 THEN H=H-1:HM=HM-1
80 IF K=0 THEN GOSUB 240
90 POKE 53248,H
100 GOTO 50
110 POKE 54279,PEEK(106)
120 PMT=PEEK(106)*256
130 FOR L=PMT TO PMT+1024:POKE L,0:NEXT L
140 P0=PMT+512
150 M0=PMT+384
160 FOR L=0 TO 11:READ D:POKE P0+L+50,D:NEXT L
170 FOR L=0 TO 1:READ D:POKE M0+L+50,D:NEXT L
180 POKE 559,46
190 POKE 53277,3
200 POKE 704,255
210 ? CHR$(125):RETURN
220 DATA 56,56,16,124,254,187,185,169,40,40,40,108
230 DATA 3,3
240 SOUND 0,F,8,10
250 POKE 53252,HM
260 IF HM<255 THEN HM=HM+1:F=F+1:GOTO 240
270 HM=H+5:F=0:SOUND 0,0,0,0:RETURN
```

40: Zusätzlich zur horizontalen Anfangsposition des Players wird die Anfangsposition des Missiles festgelegt.
50: Abfrage des Joysticks und des Feuerknopfes (Port 1).
60 und 70: Je nach Bewegung des Steuerknüppels verändert sich die Position des Players (Variable H) und des Missiles (Variable HM).
80: Sobald der Feuerknopf gedrückt wird, springt der Rechner ins Unterprogramm 240.
90: Die aktuelle Horizontalposition des Players wird gePOKEt.
150: Die Anfangsadresse für Missile 0 beginnt nach 384 Byte innerhalb der Player-Missile-Tabelle.
170: Neben den Daten für den Player werden auch die Daten für das Missile eingelesen und in der PM-Tabelle abgelegt. In diesem Fall ist das Missile zwei Linien hoch.

180 und 190: Der Bildschirm wird für die Darstellung von Playern und Missiles vorbereitet.
200: Die Farbe von Missile 0 ist identisch mit der von Player 0. Falls Sie in Ihren eigenen Programmen andersfarbige Missiles verwenden wollen, wählen Sie einfach ein anderes Missile und definieren Sie die Farbe im dazugehörenden Player-Register nach Ihren Wünschen. Alle Player und Missiles lassen sich ganz unabhängig voneinander auf dem Bildschirm darstellen und bewegen.
240: Mit diesem SOUND-Befehl läßt sich ein Schußgeräusch simulieren. Professionelle Anleitungen zur Programmierung von Soundeffekten finden Sie im "ATARI Sound- und Musik-Buch" (ebenfalls im Birkhäuser-Verlag erschienen).
250: Die aktuelle Horizontalposition des Missiles wird gePOKEt.
260: Solange HM kleiner 255 ist, nimmt sein Wert um jeweils Eins zu. Gleichzeitig erhöht sich die Frequenzangabe (Variable F) für den SOUND-Befehl.
270: Nachdem die "Kugel" aus dem Bildschirm verschwunden ist, wird der Tonkanal abgeschaltet und ins Hauptprogramm zurückgekehrt.

Nun fragen Sie sich allmählich, wann das nette Männlein endlich auch vertikal bewegt wird. (Das scheint ja eine bewegende Geschichte zu werden.) - Um es gleich zu sagen: Vertikale Player-Missile-Bewegung in BASIC ist langweilig. Im Gegensatz zur Horizontalposition, wo einfach nur der neue Wert eingePOKEt wird, müssen zur vertikalen Bewegung die Player-Daten jeweils neu in die PM-Tabelle geschrieben werden. Es entsteht durch diesen Zeitaufwand ein ungleichmäßiger Bewegungsablauf. Das folgende Listing ist daher nur der Vollständigkeit halber aufgenommen worden. Sie können diese vertikale Steuerung in BASIC nur für kleine Player einsetzen (z.B. 4 bis 8 Zeilen hoch); dabei machen sich die ungleichmäßigen Bewegungen nicht so stark bemerkbar. Für komfortable Programme sollten Sie die Maschinenspracheroutine im PM-DEMO 6 verwenden.

```
0 REM PM-DEMO 5 - VERTIKAL BASIC
10 POKE 106,PEEK(106)-4
20 GRAPHICS 5:? "MOMENT BITTE"
30 V=70:GOSUB 110
40 H=50:HM=55
50 J=STICK(0):K=STRIG(0)
60 IF J=7 AND H<250 THEN H=H+1:HM=HM+1
70 IF J=11 AND H>5 THEN H=H-1:HM=HM-1
71 IF J=14 AND V>0 THEN POKE P0+V+11,0:POKE M0+V+1,0
:V=V-1:GOSUB 280
```

```
72 IF J=13 AND V<100 THEN POKE P0+V,0:POKE M0+V,0:V=
V+1:GOSUB 280
80 IF K=0 THEN GOSUB 240
90 POKE 53248,H
100 GOTO 50
110 POKE 54279,PEEK(106)
120 PMT=PEEK(106)*256
130 FOR L=PMT TO PMT+1024:POKE L,0:NEXT L
140 P0=PMT+512
150 M0=PMT+384
160 FOR L=0 TO 11:READ D:POKE P0+L+V,D:NEXT L
170 FOR L=0 TO 1:READ D:POKE M0+L+V,D:NEXT L
180 POKE 559,46
190 POKE 53277,3
200 POKE 704,255
210 ? CHR$(125):RETURN
220 DATA 56,56,16,124,254,187,185,169,40,40,40,108
230 DATA 3,3
240 SOUND 0,F,8,10
250 POKE 53252,HM
260 IF HM<255 THEN HM=HM+1:F=F+1:GOTO 240
270 HM=H+5:F=0:SOUND 0,0,0,0:RETURN
280 RESTORE 220:FOR L=0 TO 11:READ D:POKE P0+L+V,D:N
EXT L
290 FOR L=0 TO 1:READ D:POKE M0+L+V,D:NEXT L:RETURN
```

71: Wenn der Steuerknüppel nach vorn gedrückt wird, soll sich die Vertikalposition des Players und Missiles um Eins vermindern; also auf dem Bildschirm nach oben wandern. Hierzu wird zuerst das unterste Byte des Players (P0+V+11) und Missiles (M0+V+1) gelöscht. Das ist notwendig, weil sonst die Füße des Männleins an der alten Position "stehenbleiben". Sie können das Problem auch anders lösen, indem Sie für den Player statt 12 Zeilen 14 und für das Missile 4 statt 2 Zeilen vorsehen und die oberste und unterste Zeile jeweils frei lassen (DATA-Zeile 220 und 230 um 0,...,0 ergänzen).
72: Durch Drücken des Steuerknüppels nach hinten bewegen sich Player und Missile nach unten. Hier ist es notwendig, das oberste Byte des Players und Missiles zu löschen (P0+V bzw. M0+V), da sonst der Kopf des Männleins an der alten Position "stehenbleibt". Mit dem vorgenannten Verfahren (14 Zeilen statt 12) können Sie auch hier das Problem lösen. Für welche Methode Sie sich entscheiden, bleibt Ihnen überlassen. Vor- oder Nachteile gibt es nicht.
280 und 290: Durch RESTORE können die Daten für Player und Missile neu eingelesen und in die PM-Tabelle eingetragen werden.

Mit dem folgenden Listing können Sie Player und Missile durch eine Maschinenspracheroutine vertikal bewegen. Hierbei entstehen keine unansehnlichen "Ruckelbewegungen". Die Daten für die Routine müssen sehr sorgfältig abgeschrieben werden. Fehleingaben können zum Absturz des Rechners führen. Die übrigen Programmteile sind mit dem vorhergezeigten Listing identisch.

```
0 REM PM-DEMO 6 - VERTIKAL MASCHINENSPRACHE
10 POKE 106,PEEK(106)-4
20 GRAPHICS 5:? "MOMENT BITTE":DIM VH$(21),VR$(21):V
=70
30 GOSUB 110:GOSUB 280
40 H=50:HM=55
50 J=STICK(0):K=STRIG(0)
60 IF J=7 AND H<250 THEN H=H+1:HM=HM+1
70 IF J=11 AND H>5 THEN H=H-1:HM=HM-1
71 IF J=14 AND V>0 THEN V=V-1:X=USR(VH,P0+V):X=USR(V
H,M0+V)
72 IF J=13 AND V<100 THEN POKE P0+V,0:POKE M0+V,0:V=
V+1:X=USR(VR,P0+V):X=USR(VR,M0+V)
80 IF K=0 THEN GOSUB 240
90 POKE 53248,H
100 GOTO 50
110 POKE 54279,PEEK(106)
120 PMT=PEEK(106)*256
130 FOR L=PMT TO PMT+1024:POKE L,0:NEXT L
140 P0=PMT+512
150 M0=PMT+384
160 FOR L=0 TO 11:READ D:POKE P0+L+V,D:NEXT L
170 FOR L=0 TO 1:READ D:POKE M0+L+V,D:NEXT L
180 POKE 559,46
190 POKE 53277,3
200 POKE 704,255
210 ? CHR$(125):RETURN
220 DATA 56,56,16,124,254,187,185,169,40,40,40,108
230 DATA 3,3
240 SOUND 0,F,8,10
250 POKE 53252,HM
260 IF HM<255 THEN HM=HM+1:F=F+1:GOTO 240
270 HM=H+5:F=0:SOUND 0,0,0,0:RETURN
280 VH=ADR(VH$):VR=ADR(VR$)
290 FOR L=VH TO VH+20:READ D:POKE L,D:NEXT L
300 FOR L=VR TO VR+20:READ D:POKE L,D:NEXT L:RETURN
310 DATA 104,104,133,204,104,133,203,160,1,177
320 DATA 203,136,145,203,200,200,192,17,208,245,96
330 DATA 104,104,133,204,104,133,203,160,16,177
340 DATA 203,200,145,203,136,136,192,255,208,245,96
```

Von der horizontalen und vertikalen Bewegung ist es nur ein kleiner Schritt, um Player und Missiles auch diagonal zu bewegen. Hier der Listing-Zusatz zum BASIC-Programm PM-DEMO 5:

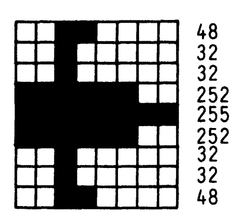
```
73 IF J=6 AND V>0 AND H<250 THEN POKE P0+V+11,0:POKE
 M0+V+1,0:V=V-1:H=H+1:HM=HM+1:GOSUB 280
74 IF J=5 AND V<100 AND H<250 THEN POKE P0+V,0:POKE
M0+V,0:V=V+1:H=H+1:HM=HM+1:GOSUB 280
75 IF J=9 AND V<100 AND H>5 THEN POKE P0+V,0:POKE M0
+V,0:V=V+1:H=H-1:HM=HM-1:GOSUB 280
76 IF J=10 AND V>0 AND H>5 THEN POKE P0+V+11,0:POKE
M0+V+1,0:V=V-1:H=H-1:HM=HM-1:GOSUB 280
```

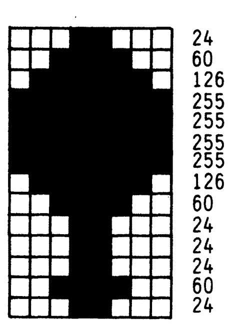
Nun noch der Listing-Zusatz zum PM-DEMO 6, bei dem die Diagonalbewegung mit der Maschinenspracheroutine durchgeführt wird:

```
70 IF J=11 AND H>5 THEN H=H-1:HM=HM-1
71 IF J=14 AND V>0 THEN V=V-1:GOSUB 350
72 IF J=13 AND V<100 THEN POKE P0+V,0:POKE M0+V,0:V=
V+1:GOSUB 360
73 IF J=6 AND V>0 AND H<250 THEN V=V-1:H=H+1:HM=HM+1
:GOSUB 350
74 IF J=5 AND V<100 AND H<250 THEN POKE P0+V,0:POKE
M0+V,0:V=V+1:H=H+1:HM=HM+1:GOSUB 360
75 IF J=9 AND V<100 AND H>5 THEN POKE P0+V,0:POKE M0
+V,0:V=V+1:H=H-1:HM=HM-1:GOSUB 360
76 IF J=10 AND V>0 AND H>5 THEN V=V-1:H=H-1:HM=HM-1:
GOSUB 350
350 X=USR(VH,P0+V):X=USR(VH,M0+V):RETURN
360 X=USR(VR,P0+V):X=USR(VR,M0+V):RETURN
```

1.6.3 Kollisionen

Im folgenden Beispielprogramm wird zunächst eine Player-Missile-Tabelle für einzeilige Auflösung eingerichtet und zwei Player definiert. Player 1 soll einen Luftballon darstellen, Player 0 eine Nadel. Die folgenden Abbildungen zeigen die Bitmuster.





(Die Nadel sieht aber komisch aus.) Ja, aber nur in dieser Abbildung. Im Programm wird Player 0 in vierfacher Breite dargestellt, so daß sie sehr langgezogen und spitz erscheint.

Mit dem Joystick können Sie auf den Ballon zufahren, wobei Sie eine gePLOTtete Wand passieren müssen (Kollision Player/Hintergrund). Sobald die Nadel ihn berührt (Kollision Player/Player), "platzt" er mit einem lauten Knall auseinander. Sie können dieses Demonstrationsprogramm selbst verändern, indem Sie andere Prioritäten setzen, weitere Player definieren oder mit Hilfe der Farbregister weitere Muster auf den Bildschirm PLOTten.

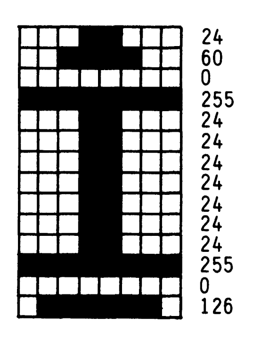
```
0 REM PM-DEMO9 KOLLISIONEN
10 POKE 106,PEEK(106)-8
20 GRAPHICS 5:? "MOMENT BITTE":COLOR 2:PLOT 30,0:DRA
WTO 30,39
30 V=70
40 GOSUB 1000
50 H0=50:H1=180
60 POKE 53249,H1
70 POKE 53248,H0
80 IF STICK(0)=7 THEN H0=H0+1
90 IF PEEK(53260)=2 THEN GOSUB 110:GOTO 50
95 IF PEEK(53252)=2 THEN GOSUB 190
100 GOTO 70
110 SOUND 0,0,8,15:POKE 53257,3
```

```
120 FOR L=0 TO 5:NEXT L
130 SOUND 0,5,8,15:POKE 53257,1
140 FOR L=0 TO 5:NEXT L
150 SOUND 0,10,8,15:POKE 53257,2
160 FOR L=0 TO 5:NEXT L
170 SOUND 0,0,0,0:POKE 53249,0
180 POKE 53278,0:FOR L=0 TO 100:NEXT L:RETURN
190 ? "Sie durchbrechen die Mauer"
200 FOR F=10 TO 100 STEP 10
210 SOUND 0,F,12,10
220 NEXT F
230 SOUND 0,0,0,0:POKE 53278,0
240 RETURN
1000 POKE 54279,PEEK(106)
1010 PMT=PEEK(106)*256
1020 FOR L=PMT+1024 TO PMT+1535:POKE L,0:NEXT L
1030 P0=PMT+1024:P1=PMT+1280
1040 FOR L=0 TO 8:READ D:POKE P0+L+V,D:NEXT L
1050 FOR L=0 TO 13:READ D:POKE P1+L+V,D:NEXT L
1060 POKE 559,62
1070 POKE 53277,3
1080 POKE 704,126
1090 POKE 705,58
1100 POKE 53256,3
1110 POKE 53257,1
1120 POKE 623,2
1130 ? CHR$(125):RETURN
1140 DATA 48,32,32,252,255,252,32,32,48
1150 DATA 24,60,126,255,255,255,255,126,60,24,24,24,
60,24
```

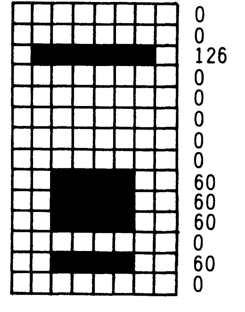
1.6.4 Zusammengefügte Player

Oft genug kommt es vor, daß die Breite von 8 Bit für eine Player-Figur nicht ausreicht oder mehrfarbige Objekte gewünscht werden. Man kann selbstverständlich mehrere Player nebeneinander anordnen, so daß sie wie eine einzige Figur erscheinen. Dabei kann jeder Player eine andere Farbe annehmen.

Eine weitere Möglichkeit zur Mehrfarbigkeit wurde bereits im Abschnitt 1.5 kurz erklärt; durch Setzen von Bit 5 (dezimal 32) im Prioritätsregister 623 erreichen Sie eine Farbüberdeckung von Player 0 und 1 sowie von Player 2 und 3. Mit dem folgenden Programmbeispiel werden Player 0 und 1 an die gleiche Position gebracht und die Wirkung einer Farbüberdeckung veranschaulicht. Die beiden Abbildungen zeigen die Bitmuster der Player.



Player 0



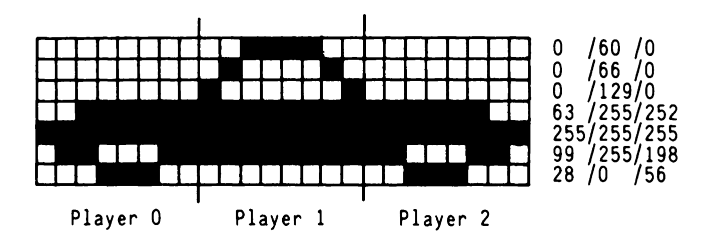
Player 1

```
0 REM PM-DEMO 10 - FARBUEBERLAGERUNG
10 POKE 106,PEEK(106)-4
20 GRAPHICS 5:? "MOMENT BITTE"
30 POKE 54279,PEEK(106)
40 PMT=PEEK(106)*256
50 FOR L=PMT+512 TO PMT+767:POKE L,0:NEXT L
60 P0=PMT+512:P1=PMT+640
70 FOR L=0 TO 13:READ D:POKE P0+L+50,D:NEXT L
80 FOR L=0 TO 13:READ D:POKE P1+L+50,D:NEXT L
90 POKE 559,42
100 POKE 53277,2
110 POKE 704,206:POKE 705,62
120 POKE 623,32
130 ? CHR$(125)
140 FOR H=40 TO 200
150 POKE 53248,H:POKE 53249,H
160 FOR L=0 TO 200:NEXT L
170 NEXT H
180 GOTO 140
190 DATA 24,60,0,255,24,24,24,24,24,24,24,255,0,126
200 DATA 0,0,126,0,0,0,0,0,60,60,60,0,60,0
```

10: Für zweizeilige Auflösung wird der erforderliche Speicherplatz reserviert.
50: Da nur zwei Player verwendet werden, muß nicht unbedingt die gesamte PM-Tabelle gelöscht werden, sondern nur der Bereich, der von den beiden Playern beansprucht wird.
60: Die Anfangsadresse von Player 0 beginnt nach 512 Byte innerhalb der PM-Tabelle; die Anfangsadresse von Player 1 nach 640 Byte.
70: Die Dezimalwerte der beiden Player werden in die Tabelle gePOKEt. Der Wert 50 legt die Vertikalposition fest. Sie können hier auch einen anderen Wert einsetzen (10 bis 115 liegt im sichtbaren Bildschirmbereich).

90 und 100: Bildschirmvorbereitung für die Darstellung von Playern (vergl. Abschnitt 1.1.1).
110: Die Farbwahl für die Player.
120: In Register 623 wird Bit 5 (Farbüberdeckung) gesetzt.
140 bis 180: Die Variable H beschreibt die Horizontalposition. Durch die FOR...NEXT-Schleife wandert die zusammengefügte und mehrfarbige Player-Figur langsam über den Bildschirm.

Im nächsten Demonstrationsbeispiel werden drei Player nebeneinander positioniert, so daß sie wie ein einziges Objekt erscheinen.




```
0 REM PM-DEMO 11 - ZUSAMMENGESTELLTE PLAYER
10 POKE 106,PEEK(106)-8
20 GRAPHICS 5:? "MOMENT BITTE"
30 POKE 54279,PEEK(106)
40 PMT=PEEK(106)*256
50 FOR L=PMT+1024 TO PMT+1791:POKE L,0:NEXT L
60 P0=PMT+1024:P1=PMT+1280:P2=PMT+1536
70 FOR L=0 TO 6:READ D:POKE P0+L+70,D:NEXT L
80 FOR L=0 TO 6:READ D:POKE P1+L+70,D:NEXT L
85 FOR L=0 TO 6:READ D:POKE P2+L+70,D:NEXT L
90 POKE 559,62
100 POKE 53277,2
110 POKE 704,90:POKE 705,90:POKE 706,90
120 POKE 623,0
130 ? CHR$(125)
140 FOR H=40 TO 200
150 POKE 53248,H:POKE 53249,H+8:POKE 53250,H+16
160 FOR L=0 TO 10:NEXT L
170 NEXT H
180 GOTO 140
190 DATA 0,0,0,63,255,99,28,60,66,129,255,255,255,0,
0,0,0,252,255,198,56
```

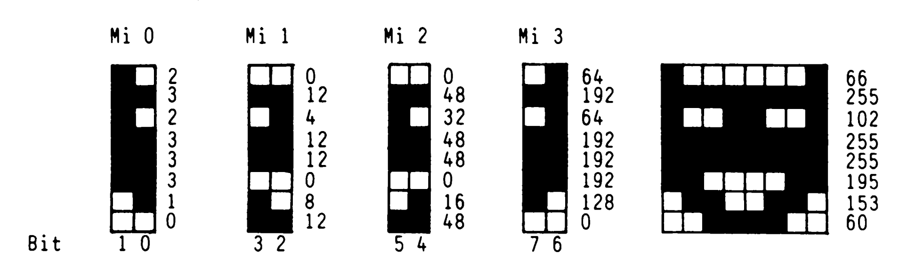
10: In diesem Programm wird PM-Grafik mit einzeiliger Auflösung gewählt. POKE 106,PEEK(106)-8 reserviert den notwendigen Speicherplatz.

50: Player 0 bis 2 nehmen bei einzeiliger Auflösung den Bereich von 1024 bis 1791 innerhalb der PM-Tabelle ein. Dieser Bereich wird zunächst gelöscht.
90: Aktivierung des Bildschirms zur Darstellung von Playern mit einzeiliger Auflösung (vergl. Abschnitt 1.1.1).
110: Festlegung der Farben für die Player.
120: Falls Sie dieses Listing aus dem vorhergehenden umschreiben, sollten Sie Register 623 wieder auf Null setzen. Wenn Sie dieses Programm jedoch ganz neu schreiben, können Sie Zeile 120 fortlassen.
150: Da die drei Player genau nebeneinander erscheinen sollen, müssen für die Horizontalposition entsprechende Angaben gemacht werden (H,H+8,H+16). Ein vierter Player müßte bei H+24 stehen. Wenn Player untereinander zusammengefügt werden sollen, müssen die Vertikalpositionen entsprechend der Höhe der Player versetzt werden. In diesem Fall stehen alle drei Player in der Vertikalposition 70 (s. Zeilen 70 bis 85).

1.6.5 Fünfter Player aus vier Missiles

Das letzte, was Sie an grundsätzlichen Dingen über Player-Missile-Grafik vielleicht schon immer einmal wissen wollten, ist das Zusammensetzen der vier Missiles zu einem fünften Player.

Hierzu setzen Sie Bit 4 (dezimal 16) in Register 623 und positionieren alle Missiles nebeneinander (H,H+2,H+4,H+6). Das zuständige Farbregister ist 711. Nur beim Entwerfen der Figur gilt es, eine Spielregel zu beachten: Im Gegensatz zur Darstellung in der PM-Tabelle müssen Sie die Missiles spiegelverkehrt anordnen; also zuerst Missile 0 (Bit 1+0), dann Missile 1 (Bit 3+2), Missile 2 (Bit 5+4) und schließlich Missile 3 (Bit 7+6). Die Dezimalwerte, die Sie neben der vollständigen Player-Abbildung sehen, haben also nichts mit dem üblichen Bitmuster zu tun (die oberste Zeile müsste dann den Wert 129 tragen); sie stellen sich vielmehr aus der Addition der einzelnen Missile-Daten zusammen (z.B. oberste Zeile 2+64=66). Warum diese hardwaremäßige Verkomplizierung



bei ATARI vorgenommen wurde, wird wohl immer eines der ungeklärten Rätsel der Menschheit bleiben.

```
0 REM PM-DEMO 12 - MISSILES ALS PLAYER
10 POKE 106,PEEK(106)-4
20 GRAPHICS 5:? "MOMENT BITTE"
30 POKE 54279,PEEK(106)
40 PMT=PEEK(106)*256
50 FOR L=PMT+384 TO PMT+511:POKE L,0:NEXT L
60 MP=PMT+384
70 FOR L=0 TO 7:READ D:POKE MP+L+50,D:NEXT L
90 POKE 559,38
100 POKE 53277,1
110 POKE 711,138
120 POKE 623,16
130 ? CHR$(125)
140 FOR H=40 TO 200
150 POKE 53252,H:POKE 53253,H+2:POKE 53254,H+4:POKE
53255,H+6
160 FOR L=0 TO 10:NEXT L
170 NEXT H
180 GOTO 140
190 DATA 66,255,102,255,255,195,153,60
```

50: Es wird nur der Bereich gelöscht, den die Missiles einnehmen.
90 und 100: Darstellung Missiles bei zweizeiliger Auflösung (vergl. Abschnitt 1.1.1).
120: Bit 4 in Register 623 zur Darstellung der vier Missiles als fünften Player.
140 bis 180: Durch die FOR...NEXT-Schleife wandert die Maske über den Bildschirm. Sie können diesen Teil auch fortlassen und den fünften Player an einer festen Horizontalposition erscheinen lassen.

2. Kapitel

Anwendungen

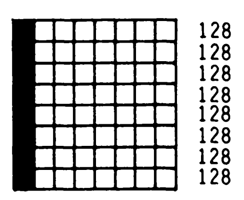
2.1 Player als "Maus"

Die Anwendungsmöglichkeiten für Player-Missile-Grafik sind geradezu unerschöpflich. (Das schreiben immer die Autoren, denen nichts besseres einfallt.) In diesem Fall stimmt es aber. Denn im Gegensatz zu der Bezeichnung lassen sich Player und Missiles nicht nur als "Spieler" und "Geschosse" einsetzen. Im folgenden KARTEI-Programm wird z.B. ein Player als Cursor benutzt, mit dem man wie mit einer "Maus" auf dem Bildschirm herumfahren kann, um bestimmte Funktionen aufzurufen oder Texte einzutragen.

Anstelle der externen Steuerung erfolgt hier die Steuerung über die vier Pfeiltasten. Durch Drücken von RETURN können Sie eine Eintragung machen oder eine Funktion aufrufen (z.B. Karte speichern). Die Verwendung eines Players hat den großen Vorteil, daß bei der Bewegung auf dem Bildschirm nicht ständig die vor und neben einem Cursor befindlichen Zeichen gelöscht und wieder aufgerufen werden müssen. Der Player bewegt sich einfach über die Buchstaben hinweg.

Mit dem KARTEI-Programm können Sie Daten jeder Art archivieren; z.B. Bücher, Videofilme oder eine Briefmarkensammlung. Das Listing ist für den Betrieb mit Diskettenstation ausgearbeitet; falls Sie einen Datenrecorder benutzen, müssen Sie die Befehle zur Datenübertragung entsprechend ändern (z.B. beim Dateinamen C: statt D: einsetzen). Das bereitet jedoch keine Probleme.

Der Player ist 8 Zeilen hoch, wobei in jedem Byte Bit 7 (dezimal 128) gesetzt wird. Im Programm erscheint der Player dann in vierfacher Breite, so daß Sie einen schönen Cursor vor Augen haben.



```
0 REM KARTEI
10 POKE 106,PEEK(106)-8
20 GRAPHICS 0:DIM VH$(21),VR$(21),DN$(14),X$(14),DT$
(10),STW$(9),NR$(10)
30 DIM E1$(30),E2$(30),E3$(30),E4$(30)
40 DN$="D:":V=40:HP=72
50 POKE 752,1:POKE 710,146
100 GOSUB 1000:GOSUB 1200
110 J=PEEK(764)
120 IF J=7 AND HP<240 THEN HP=HP+4
130 IF J=6 AND HP>4 THEN HP=HP-4
140 IF J=14 AND V>4 THEN V=V-1:X=USR(VH,P0+V)
150 IF J=15 AND V<240 THEN POKE P0+V,0:V=V+1:X=USR(V
R,P0+V)
160 IF J=12 THEN POKE 764,255:GOSUB 1400
200 POKE 53248,HP
300 IF PEEK(753)=0 THEN POKE 764,255:FOR L=0 TO 80:N
EXT L
305 XH=HP:XV=V
310 GOTO 110
1000 POKE 54279,PEEK(106):PMT=PEEK(106)*256
1010 FOR L=PMT+1024 TO PMT+1279:POKE L,0:NEXT L:P0=P
MT+1024
1020 POKE 559,58:POKE 53277,2:POKE 704,206:POKE 623,
1:POKE 53256,3
1030 FOR L=0 TO 7:READ D:POKE P0+L+V,D:NEXT L
1040 DATA 128,128,128,128,128,128,128,128
1100 VH=ADR(VH$):VR=ADR(VR$)
1110 FOR L=VH TO VH+20:READ D:POKE L,D:NEXT L
1120 FOR L=VR TO VR+20:READ D:POKE L,D:NEXT L:RETURN

1130 DATA 104,104,133,204,104,133,203,160,1,177
1140 DATA 203,136,145,203,200,200,192,17,208,245,96
1150 DATA 104,104,133,204,104,133,203,160,16,177
1160 DATA 203,200,145,203,136,136,192,255,208,245,96
1200 REM KARTEIKARTE
1210 POSITION 0,1:? "Datei:";X$
1220 POSITION 21,1:? "Kartei-Nr.:";NR$
1230 POSITION 0,3:? "Stichwort:";STW$
1240 POSITION 21,3:? "Datum:";DT$
1250 POSITION 0,4:? "______________________________
_________"
1260 POSITION 2,6:? "1. Eintrag:":POSITION 1,7:? E1$
1270 POSITION 2,9:? "2. Eintrag:":POSITION 1,10:? E2
$
1280 POSITION 2,12:? "3. Eintrag:":POSITION 1,13:? E
3$
```

```
1290 POSITION 2,15:? "4. Eintrag:":POSITION 1,16:? E
4$
1300 POSITION 0,18:? "______________________________
_________"
1310 POSITION 2,20:? "* Karte loeschen   * Karte dru
cken"
1320 POSITION 2,21:? "* Karte speichern  * Karte ein
lesen"
1330 POSITION 2,22:? "* Datei einlesen   * Datei dru
cken"
1340 POSITION 2,23:? "* Disketteninhalt  * Neue Date
i";
1350 RETURN
1400 REM EINGABEFUNKTIONEN
1410 IF HP=72 AND V<45 THEN POSITION 7,1:INPUT X$:DN
$(3)=X$:POSITION 7,1:? " ";
1420 IF HP=176 AND V<45 THEN POSITION 32,1:INPUT NR$
:POSITION 32,1:? " ";
1430 IF HP=88 AND V<60 THEN POSITION 11,3:INPUT STW$
:POSITION 11,3:? " ";
1440 IF HP=156 AND V<60 THEN POSITION 28,3:INPUT DT$
:POSITION 28,3:? " ";
1450 IF HP=100 AND V>70 AND V<90 THEN POSITION 1,7:I
NPUT E1$:POSITION 1,7:? " ";
1460 IF HP=100 AND V>94 AND V<114 THEN POSITION 1,10
:INPUT E2$:POSITION 1,10:? " ";
1470 IF HP=100 AND V>118 AND V<138 THEN POSITION 1,1
3:INPUT E3$:POSITION 1,13:? " ";
1480 IF HP=100 AND V>142 AND V<162 THEN POSITION 1,1
6:INPUT E4$:POSITION 1,16:? " ";
1490 IF HP=56 AND V>187 AND V<197 THEN GOSUB 1590:?
CHR$(125):GOTO 1200
1500 IF HP=132 AND V>187 AND V<197 THEN GOSUB 1600
1510 IF HP=56 AND V>195 AND V<205 THEN GOSUB 1700
1520 IF HP=132 AND V>195 AND V<205 THEN GOSUB 1800
1530 IF HP=56 AND V>203 AND V<213 THEN GOSUB 1900
1540 IF HP=132 AND V>203 AND V<213 THEN GOSUB 2000
1550 IF HP=56 AND V>211 AND V<221 THEN GOSUB 2100
1560 IF HP=132 AND V>211 AND V<221 THEN GOSUB 2200
1570 RETURN
1590 NR$="":STW$="":DT$="":E1$="":E2$="":E3$="":E4$=
"":RETURN
1600 REM KARTE DRUCKEN
1610 LPRINT "Datei: ";DN$;"  Kartei-Nr.: ";NR$:LPRIN
T
1620 LPRINT "Stichwort: ";STW$;"  Datum: ";DT$:LPRIN
T
```

```
1630 LPRINT "----------------------------------------
-"
1640 LPRINT "1. Eintrag:":LPRINT E1$:LPRINT
1650 LPRINT "2. Eintrag:":LPRINT E2$:LPRINT
1660 LPRINT "3. Eintrag:":LPRINT E3$:LPRINT
1670 LPRINT "4. Eintrag:":LPRINT E4$:LPRINT
1680 LPRINT "----------------------------------------
-"
1690 LPRINT :LPRINT :RETURN
1700 REM KARTE SPEICHERN
1710 POSITION 2,22:? CHR$(156);:? CHR$(156);
1720 POSITION 2,22:? "Neue Datei (8) oder Anhang (9)
";
1730 INPUT P:IF ASC(DN$(3))>32 THEN OPEN #1,P,0,DN$
1740 IF ASC(DN$(3))=32 THEN POSITION 2,23:? "Eingabe
 Dateiname: ";:INPUT X$:DN$(3)=X$:OPEN #1,P,0,DN$
1750 X$=CHR$(155)
1760 PRINT #1;NR$;X$;STW$;X$;DT$;X$;E1$;X$;E2$;X$;E3
$;X$;E4$;X$
1770 CLOSE #1
1780 GOTO 1310
1800 REM KARTE EINLESEN
1810 POSITION 2,22:? CHR$(156);:? CHR$(156);
1820 POSITION 2,22:? "Eingabe Kartei-Nr.:";:INPUT NR
1830 IF ASC(DN$(3))=32 THEN POSITION 2,23:? "Eingabe
 Dateiname: ";:INPUT X$:DN$(3)=X$
1840 TRAP 1880:OPEN #1,4,0,DN$
1850 INPUT #1,NR$,STW$,DT$,E1$,E2$,E3$,E4$
1860 IF VAL(NR$)=NR THEN CLOSE #1:GOTO 1200
1870 GOTO 1850
1880 CLOSE #1:GOTO 1200
1900 REM DATEI EINLESEN
1910 POSITION 2,22:? CHR$(156);:? CHR$(156);:OPEN #2
,4,0,"K:"
1920 IF ASC(DN$(3))=32 THEN POSITION 2,23:? "Eingabe
 Dateiname: ";:INPUT X$:DN$(3)=X$
1930 TRAP 1990:OPEN #1,4,0,DN$
1935 INPUT #1,NR$,STW$,DT$,E1$,E2$,E3$,E4$
1940 GOSUB 1220
1950 POSITION 2,22:? CHR$(156);:? CHR$(156);
1960 POSITION 2,22:? "Weiterlesen mit W  Ende mit E"
;:GET #2,A
1970 IF A=69 THEN CLOSE #1:CLOSE #2:GOTO 1310
1980 GOTO 1935
1990 CLOSE #1:CLOSE #2:GOTO 1310
2000 REM DATEI DRUCKEN
2010 POSITION 2,22:? CHR$(156);:? CHR$(156);
```

```
2020 IF ASC(DN$(3))=32 THEN POSITION 2,22:? "Eingabe
 Dateiname: ";:INPUT X$:DN$(3)=X$
2030 TRAP 2070:OPEN #1,4,0,DN$
2040 INPUT #1,NR$,STW$,DT$,E1$,E2$,E3$,E4$
2050 GOSUB 1600
2060 GOTO 2040
2070 CLOSE #1:GOTO 1310
2100 REM DISKETTENINHALT
2110 TRAP 2130:OPEN #1,6,0,"D:*.*":? CHR$(125)
2120 INPUT #1,X$:? X$:GOTO 2120
2130 CLOSE #1:? :? "Weiter mit RETURN"
2140 OPEN #1,4,0,"K:"
2150 GET #1,A:IF A=155 THEN CLOSE #1:? CHR$(125):X$=
"":GOTO 1200
2200 REM NEUE DATEI
2210 CLR :RUN
```

10: Es wird PM-Grafik mit einzeiliger Auflösung gewählt.
20 und 30: Die Variablen für die einzelnen Eintragungen (Dateiname, Datum, Stichwort, Karteinummer) werden DIMensioniert.
40: V und HP sind die Variablen für die vertikale und horizontale Position des Players.
50: POKE 752,1 macht den normalen Cursor unsichtbar (Normalwert = 0). Mit Register 710 bestimmen Sie die Hintergrundfarbe des Bildschirms.
110: 764 ist das Tastaturregister. Die Werte, die hier je nach Tastendruck gespeichert werden, haben nichts mit dem ASCII-Code zu tun; es handelt sich um interne Werte.
120 bis 150: Je nachdem, welche Pfeiltaste Sie drücken, bewegt sich der Player horizontal oder vertikal.
160: Wenn Sie die RETURN-Taste drücken, (Wert 12 im Tastaturregister), können Sie entweder eine Texteingabe machen oder eine Funktion aufrufen (s. 1310 bis 1340). Zur Texteingabe positionieren Sie den Player immer hinter den Doppelpunkt der gewünschten Eingabe (z.B. Stichwort:). Um eine Funktion aufzurufen, setzen Sie den Player immer über das Sternchen vor der gewünschten Funktion (z.B. * Karte löschen). In den Abfragen des Programms wurden Toleranzwerte von -5 bis +5 über bzw. unter der betreffenden Bildschirmzeile eingearbeitet, so daß Sie den Player nicht millimetergenau steuern müssen.
200: Die aktuelle Horizontalposition des Players.
300: Der Wert des Tastaturregisters bleibt bis zum Drücken einer neuen Taste erhalten. Mit Register 753 kann festgestellt werden, ob überhaupt eine Taste gedrückt wird. Wenn dies nicht der Fall ist (Wert = 0), dann wird das Tasteturregister durch POKE 764,255 gelöscht.

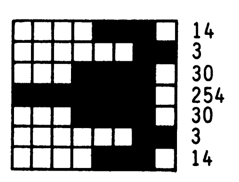
1000 bis 1160: Die Player-Missile-Tabelle, Daten für die Farbe, Maschinenspracheprogramm zur vertikalen Bewegung usw. wie es in den vorhergehenden Demo-Programmen bereits ausführlich kommentiert wurde.
1200 bis 1300: Hier wird die Karteikarte auf den Bildschirm gebracht, in die Sie Ihre Eintragungen vornehmen können. Wenn Sie z.B. ein Videoarchiv einrichten, können Sie als "Stichwort" das Genre (Krimi, Western usw.) eingeben, unter "1. Eintrag" den Regisseur oder die Videofirma angeben usw. Das Dateiprogramm ist also ganz universell einsetzbar.
1310 bis 1340: Hier werden die Funktionen auf den Bildschirm gebracht, die Sie mit der "Maus" aufrufen können: den Text der aktuellen Karteikarte löschen, eine einzelne Karte ausdrucken, einlesen oder abspeichern, Einlesen oder Ausdrucken einer ganzen Datei, das Inhaltsverzeichnis einer Diskette auflisten und eine neue Datei erstellen. Wie oben bereits beschrieben, müssen Sie den Player nur über das Sternchen vor der Funktion plazieren und RETURN drücken.
1410: Wenn Sie mit dem Player hinter die Doppelpunkte des ersten Textes der Karteikarte fahren ("Datei:"), ist die Horizontalposition des Players = 72 und die Vertikalposition = 40. Sobald Sie RETURN drücken, erscheint das Fragezeichen, das immer bei INPUT-Befehlen auftritt. Sie können nun den Dateinamen eingeben (z.B. VIDEOT.HEK). Wenn Sie wieder RETURN drücken, wird der Dateiname registriert und das Fragezeichen gelöscht. Sie können jetzt den Player weiter bewegen.
1420 bis 1490: Genau wie für "Datei:" können Sie auch hinter den Doppelpunkten der anderen Texte Eintragungen vornehmen, wenn Sie den Player dort plazieren und RETURN drücken.
1500 bis 1570: Hier wird, je nach Position des Players, in das Unterprogramm gesprungen, das beim Aufruf einer Funktion ausgeführt werden soll.
1600 bis 1690: Ausgabe der Karteikarte an den Drucker.
1710 und 1720: Falls Sie zum ersten Mal eine Karteikarte speichern wollen, also eine Datei eröffnen, müssen Sie als zweiten Parameter des OPEN-Befehls eine 8 eingeben. Wenn Sie hingegen an eine bereits bestehende Datei anhängen wollen, ist der zweite Parameter eine 9. Mit Zeile 1710 werden die untersten beiden Bildschirmzeilen gelöscht und die Abfrage aus Zeile 1720 an diese Stelle gePRINTet.
1730: Nur wenn der ASCII-Wert des dritten Zeichens des Dateinamens (D:x...) kein Leerzeichen ist (ASCII-Wert = 32), wird die Datei zur Diskettenstation eröffnet. Diese Sicherung wurde eingebaut, falls Sie vergessen, einen Dateinamen zu definieren.
1750: Um die einzelnen Datensätze zu trennen, muß hinter jedem Eintrag ein EOL (End of Line) eingegeben werden.

1760: Die Eintragungen der Karteikarte werden an die Diskettenstation abgegeben.
1800 bis 1880: Mit dem KARTEI-Programm haben Sie die Möglichkeit, einzelne Karteikarten aus der Gesamtdatei zu selektieren. Hierzu geben Sie die Karteinummer ein, die vom Rechner gesucht und auf dem Bildschirm ausgegeben wird. Sollte sich die gesuchte Karte nicht in der Datei befinden, wird der Selektiervorgang einfach abgebrochen. Natürlich können Sie auch nach anderen Kriterien auswählen; z.B. indem Sie die Datei nach Stichworten durchsuchen. Zeile 1860 müßte dann lauten: IF STW$=SUCHWORT$ THEN CLOSE#1:GOTO 1200. (SUCHWORT$ müßten Sie in Zeile 30 noch DIMensionieren.)
1900 bis 1990: Anstelle einer einzelnen Karte (Unterprogramm 1800) wird hier die gesamte Datei eingelesen. Der TRAP-Befehl in Zeile 1930 sorgt dafür, daß ins Hauptprogramm zurückgesprungen wird, sobald eine Lesefehler auftritt (Dateiende).
2000 bis 2070: Anstelle einer einzelnen Karte (Unterprogramm 1600) wird hier eine gesamte Datei auf dem Drucker ausgegeben.
2100 bis 2150: Mit dem Parameter 6 des OPEN-Befehls kann man das Inhaltsverzeichnis einer Diskette lesen, ohne DOS aufzurufen. Diese Funktion ist hier nützlich, wenn Sie z.B. wissen wollen, wieviel Speicherplatz sich noch auf einer Diskette befindet oder wenn Sie prüfen wollen, welche Dateinamen schon existieren.

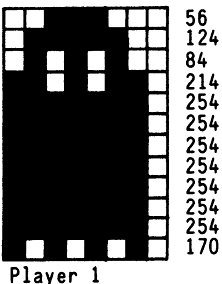
2.2 Player-Missile-Grafik als Spielelemente

Neben der "seriösen" Anwednung, wie im vorhergehenden Programm gezeigt, läßt sich PM-Grafik freilich auch für Computerspiele einsetzen. Die Player eignen sich dabei als Figuren, Gefäße, Raumschiffe, Monster usw.; die Missiles als Geschosse oder für einen fünften Player (vergl. Abschnitt 1.6.5).

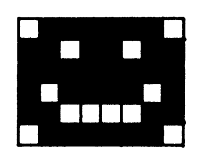
Falls Sie einmal nach Herzenslust herumballern wollen, können Sie das folgende Programm eingeben, bei dem es um eine "Geisterjagd" geht: Ständig tauchen an zufälligen Stellen Geister mit grinsenden Gesichtern oder Finstermienen in allen verfügbaren Farben auf und hetzen auf Sie zu. Mit einer Kanone können Sie sich der Unholde erwehren. Das Spiel ist gleichzeitig ein schönes PM-Beispiel, da alle vier Player eingesetzt werden und die Farben und Positionen der Figuren ständig wechseln. Selbst wenn Sie die Geister nicht "abschießen" wollen, können Sie sich allein durch den farbenprächtigen Anblick unterhalten lassen.



Player 0



Player 1



Player 2

56
68
186
146
146
130
186
198
124
68
68
68

Player 3

```
0 REM BALLER
10 POKE 106,PEEK(106)-4
20 GRAPHICS 1:? "EIN MOMENT BITTE"
30 DIM VH$(21),VR$(21),F(2),H(2),V(2)
40 FOR Z=0 TO 2:V(Z)=INT(80*RND(0))+10:NEXT Z
50 H=190:V=50:VM=V+3:R=53256:GOSUB 10000
60 FOR Z=0 TO 2:F(Z)=INT(16*RND(0))*16+12:NEXT Z
70 FOR Z=0 TO 2:H(Z)=INT(100*RND(0))+30:NEXT Z
80 Z=0
100 POKE 705,F(0):POKE 706,F(1):POKE 707,F(2)
110 J=STICK(0):K=STRIG(0)
120 IF J=14 THEN V=V-1:VM=VM-1:X=USR(VH,P0+V):X=USR(
VH,MI+VM)
130 IF J=13 THEN POKE P0+V,0:POKE MI+VM,0:V=V+1:VM=V
M+1:X=USR(VR,P0+V):X=USR(VR,MI+VM)
140 POKE 53249,H(0):POKE 53250,H(1):POKE 53251,H(2):
POKE 53252,H+3
150 IF H(0)<220 THEN H(0)=H(0)+1
160 IF H(1)<220 THEN H(1)=H(1)+1
170 IF H(2)<220 THEN H(2)=H(2)+1
180 IF K=0 THEN KX=1
190 IF KX=1 THEN H=H-10:F=F+1:SOUND 0,F,8,15
200 IF H<45 THEN SOUND 0,0,0,0:H=190:F=0:KX=0
```

```
210 IF H(0)=220 THEN H(0)=INT(100*RND(0))+30:F(0)=IN
T(16*RND(0))*16+12
220 IF H(1)=220 THEN H(1)=INT(100*RND(0))+30:F(1)=IN
T(16*RND(0))*16+12
230 IF H(2)=220 THEN H(2)=INT(100*RND(0))+30:F(2)=IN
T(16*RND(0))*16+12
240 IF PEEK(R)<>1 THEN POKE 53278,0:FOR Y=0 TO 10:NE
XT Y:GOSUB 500
250 IF PEEK(53260)<>0 THEN ? "VERLOREN":END
300 GOTO 100
500 REM KOLLISION
501 IF PEEK(R)=2 THEN Z=1
502 IF PEEK(R)=4 THEN Z=2
503 IF PEEK(R)=8 THEN Z=3
504 IF Z>3 THEN RETURN
510 SOUND 0,0,8,10:POKE R+Z,1
520 SOUND 0,5,8,10:POKE R+Z,2
530 SOUND 0,10,8,10:POKE R+Z,1
540 SOUND 0,15,8,10:POKE R+Z,2
550 SOUND 0,20,8,10:POKE R+Z,3
560 SOUND 0,5,8,10:POKE R+Z,2
570 SOUND 0,0,0,0:PU=PU+1:? CHR$(125);PU
580 Z=Z-1:POKE 53249+Z,0
590 V(Z)=INT(80*RND(0))+10:F(Z)=INT(16*RND(0))*16+12
:H(Z)=INT(100*RND(0))+30
600 RESTORE 10110+Z*10
610 IF Z=0 THEN Z=11:PL=P1:X=0
620 IF Z=1 THEN Z=5:PL=P2:X=1
630 IF Z=2 THEN Z=11:PL=P3:X=2
640 FOR L=0 TO 127:POKE PL+L,0:NEXT L
650 FOR L=0 TO Z:READ D:POKE PL+L+V(X),D:NEXT L
660 H=190:POKE 53252,H+3
670 POKE 53278,0:X=0:F=0:KX=0
680 RETURN
10000 REM PM-TABELLE
10010 POKE 54279,PEEK(106)
10020 PMT=PEEK(106)*256
10030 FOR L=PMT TO PMT+1024:POKE L,0:NEXT L
10040 P0=PMT+512:P1=P0+128:P2=P1+128:P3=P2+128:MI=PM
T+384
10050 FOR L=0 TO 6:READ D:POKE P0+L+V,D:NEXT L
10060 FOR L=0 TO 11:READ D:POKE P1+L+V(0),D:NEXT L
10070 FOR L=0 TO 5:READ D:POKE P2+L+V(1),D:NEXT L
10080 FOR L=0 TO 11:READ D:POKE P3+L+V(2),D:NEXT L
10090 POKE MI+VM,3:POKE MI+1+VM,3
10100 DATA 14,3,30,254,30,3,14
10110 DATA 56,124,84,214,254,254,254,254,254,254,254
```

```
,170
10120 DATA 126,219,255,189,195,126
10130 DATA 56,68,186,146,146,130,186,198,124,68,68,6
8
10210 POKE 559,46:POKE 53277,3:POKE 623,2
10220 POKE 704,58:POKE 53248,H:POKE 53260,1
10230 ? CHR$(125)
10300 VH=ADR(VH$):VR=ADR(VR$)
10310 FOR L=VH TO VH+20:READ D:POKE L,D:NEXT L
10320 FOR L=VR TO VR+20:READ D:POKE L,D:NEXT L:RETUR
N
10330 DATA 104,104,133,204,104,133,203,160,1,177
10340 DATA 203,136,145,203,200,200,192,17,208,245,96
10350 DATA 104,104,133,204,104,133,203,160,16,177
10360 DATA 203,200,145,203,136,136,192,255,208,245,9
6
```

40: Die Vertikalpositionen der Geister werden in jedem Durchlauf zufällig erzeugt.
50: Die Anfangsposition der Kanone.
60 und 70: Zufallsermittlung für die Farben und Horizontalpositionen der Geister.
100: Die aktuellen Farben der Geister werden gePOKEt.
110: Die Bewegung der Kanone erfolgt über Joystick (Port 1).
120 und 130: Je nach Bewegung des Steuerknüppels bewegt sich die Kanone vertikal auf oder ab.
140: Die aktuellen Horizontalpositionen der Geister.
150 bis 170: Solange die Geister den sichtbaren Bildschirmbereich noch nicht verlassen haben (Wert ab 220), sollen sie sich von links nach rechts bewegen (Hx=Hx+1).
180: Wenn der Feuerknopf gedrückt wird (K=0), nimmt die Hilfsvariable KX den Wert 1 an.
190: Wenn HX=1 ist, soll sich Missile 0 in Zehnerschritten auf die Geister zubewegen. Gleichzeitig ertönt ein Schußgeräusch (Einzelheiten über Soundprogrammierung in "ATARI Sound- und Musik-Buch", Birkhäuser-Verlag).
200: Sobald das Missile aus dem sichtbaren Bildschirmbereich wandert (H kleiner 45), was bedeutet, daß kein Geist getroffen wurde, wird der Tonkanal abgeschaltet und das Missile wieder der Position der Kanone zugeordnet.
210 bis 230: Sobald ein Geist aus dem Bildschirmfenster gleitet, wird seine Vertikal- und Horizontalposition sowie seine Farbe neu bestimmt. Er erscheint dann wieder auf dem Bildschirm. Das erweckt den Eindruck, als tauchten ständig neue Geister "aus dem Nichts" auf.

240: 53256 ist das Kollisionsregister für Missile 0 (Variable R). Wenn eine Kollision mit einem Geist erfolgt, nimmt das Register den Wert 2,4 oder 8 an (vergl. Abschnitt 1.4). In diesem Fall springt der Rechner ins Unterprogramm 500.
250: Sobald ein Geist mit der Kanone kollidiert, ist das Spiel verloren.
510 bis 580: Wird ein Geist getroffen, verschwindet er mit einem Knall vom Bildschirm. Als besonderer Effekt wird das Register für die Breite des betroffenen Players kurz auf doppelte und vierfache Breite geschaltet.
590: Für den getroffenen Geist wird eine neue Position und Farbe errechnet.
600: In den Zeilen 10110 bis 10130 stehen die Daten für die drei Geister. Für den getroffenen Geist werden die Daten RESTOREd; d.h. sie können erneut gelesen werden. Wenn Geist 2 getroffen wurde (Player 2), ist Z=1. Somit wird 10110+1*10= 10120 RESTOREd.
610 bis 630: Da die Geister unterschiedlich hoch sind, muß noch festgelegt werden, wieviele FOR...NEXT-Durchläufe zum Dateneinlesen notwendig sind.
640 und 650: Zunächst wird der Tabellenbereich des betroffenen Players gelöscht (Sie sehen sonst zwei identische Player auf dem Bildschirm), danach die Daten an die neue Vertikalposition gePOKEt.
660: Das Missile wird auf die ursprüngliche Position (Kanone) zurückgesetzt.
670: Löschen des Kollisionsregisters.
10000 bis 10360: Für dieses Spiel wurde Player-Missile-Grafik mit zweizeiliger Auflösung gewählt. Der Aufbau der PM-Tabelle wurde im ersten Kapitel des Buches ausführlich erklärt.

2.3 Nützliche Bildschirmroutinen

Die folgenden Abschnitte haben direkt nichts mit Player-Missile-Grafik zu tun; es handelt sich um Programmiertricks für Bildschirmdarstellungen. Sicher haben Sie schon einmal in professionellen Programmen beobachtet, daß die Grafik des Bildschirms blitzschnell wechselt oder ganze Landschaften vorüberrollen. Solche Zaubereien können Sie auch in Ihre eigenen Programme einarbeiten. Sie gewinnen dadurch die Möglichkeit, neben der farbenreichen und feinauflösenden Player-Missile-Grafik weitere interessante Effekte zu verwirklichen.

2.3.1 Page-Flipping

Unter "Page-Flipping" ist das blitzschnelle Wechseln von Bildschirmseiten zu verstehen. Page-Flipping kann z.B. eingesetzt werden, wenn ein Player auf dem Bildschirm eine Tür berührt und im nächsten Moment das Innere des Raumes erscheint. Genauso gut lassen sich aber auch einzelne Seiten eines geschriebenen Textes auf diese Weise aufrufen.

Um Page-Flipping zu erreichen, muß der "Zeiger" auf die sog. Display-Liste geändert werden. In dieser Liste sind die Daten für den Bildschirmaufbau enthalten sowie die Adressen für den Anfang des Bildschirmspeichers und der Display-Liste selbst. Die gesamte Thematik ist relativ kompliziert; in dem Buch "ATARI-BASIC Trickkiste" (Birkhäuser-Verlag) wird die Display-Liste in einem eigenen Kapitel ausführlich erklärt. Für das folgende Beispiel daher nur das Wichtigste: Wenn Sie z.B. GRAPHICS 3+16 einschalten, stehen Ihnen 40 Spalten und 24 Zeilen zur Verfügung. In den Adressen 560 und 561 stehen das Low-Byte und High-Byte für den Anfang der Display-Liste (Formel PEEK(560)+PEEK(561)*256 = Anfang). Wenn Sie sich von dieser Anfangsadresse aus die Inhalte der 32 folgenden Adressen ausdrucken lassen, erhalten Sie für GRAPHICS 3+16 folgende Display-Liste:

```
112     Leerzeile
112     Leerzeile
112     Leerzeile
72      GR.3-Zeile (8) + Option der Anfangsadresse
        für den Bildschirmspeicher (64)
112     Low-Byte Bildschirmspeicher
158     High-Byte Bildschirmspeicher
8       23 verbleibende GR.3-Zeilen
8
```

8
8
8
8
8
8
8
8
8
8
8
8
8
8
8
8
8
8
8
8
8
65 Rücksprungbefehl zu:
80 Low-Byte Display-Liste
158 High-Byte Display-Liste

Diese Zahlen wurden auf dem 800XL ermittelt; die Low-Byte- und High-Byte-Werte können daher bei den anderen Modellen verschieden sein. Mit der 33. Adresse hinter dem Anfang der Display-Liste beginnt also der Bildschirmspeicher (in diesem Fall die Adresse 40560). Dies ist die oberste linke Ecke des Bildschirms in GRAPHICS 3+16. Für jede Grafikbetriebsart gibt es eine andere Display-Liste, die Sie sich nach der oben gezeigten Methode auflisten können. Eine Zeile in GRAPHICS 0 (beim Einschalten) hat z.B. den Wert 2. Sie können zum Spaß einmal mitten in die GRAPHICS-Zeilen eine Leerzeile (112) POKEn. Dann erhalten Sie einen unterbrochenen Bildschirm mit schwarzer Linie.

Beim Page-Flipping werden Display-Listen in einem reservierten Speicherbereich abgelegt und nach Bedarf aufgerufen. Dazu müssen Sie im Programmablauf nur die Anfangsadresse der aktuellen Display-Liste in Register 561 (High-Byte) einPOKEn.

```
0 REM PAGE-FLIPPING
10 GRAPHICS 3+16
20 A=PEEK(561)
30 COLOR 1
40 PLOT 0,10
50 DRAWTO 39,10
60 PLOT 0,13
70 DRAWTO 39,13
80 POKE 106,PEEK(106)-2
90 GRAPHICS 3+16
100 B=PEEK(561)
110 COLOR 2
120 PLOT 0,15
130 DRAWTO 39,10
140 PLOT 0,18
150 DRAWTO 39,13
160 POKE 561,B
170 FOR X=0 TO 1000
180 NEXT X
190 POKE 561,A
200 FOR X=0 TO 1000
210 NEXT X
220 GOTO 160
```

10: Die Grafikbetriebsart 3+16 wird gewählt.
20: Das High-Byte für den Anfang der Display-Liste wird der Variable A zugewiesen.
30 bis 70: Es werden zwei Linien auf den Bildschirm gezeichnet.
80: Für die zweite Bildschirmseite muß vom RAMTOP aus (oberste Adresse des vefügbaren RAMs) ein Speicherbereich reserviert werden, in dem die Display-Liste und der Bildschirmspeicher Platz haben. GRAPHICS 3+16 beansprucht hierfür 432 Byte. Der Einfachheit halber sollten Sie immer ganze "Seiten" (=256 Byte) vom RAMTOP aus reservieren. In diesem Fall genügen also zwei Seiten (=512 Byte), um eine Display-Liste für GRAPHICS 3+16 unterzubringen.
90: Durch erneutes Aufrufen von GRAPHICS 3+16 kann die zweite Bildschirmseite gestaltet werden.
100: Die hierfür zuständige Anfangsadresse der Display-Liste wird der Variable B zugewiesen.
110 bis 150: Auch auf die zweite Bildschirmseite werden Linien gezeichnet.
160 bis 220: Beide Display-Listen werden durch die abwechselnd durch POKE 561,A und POKE 561,B aufgerufen. Dadurch erscheinen die verschiedenen Linienmuster blitzschnell auf dem Bildschim.

Nach diesem Prinzip können Sie alles mögliche auf die einzelnen Bildschirmseiten zeichnen (z.B. ein Labyrinth, das sich über mehrere Seiten ausdehnt). Wenn Sie andere Grafikbetriebsarten verwenden, müssen Sie darauf achten, daß Sie genügend Speicherplatz für jede Display-Liste reservieren. GRAPHICS 5+16 z.B. beansprucht wegen der feineren Auflösung 1176 Byte. Hier müssen Sie also 5 Seiten vom RAMTOP aus reservieren (5*256=1280). Dem Page-Flipping sind somit natürliche Grenzen gesetzt, da nicht endlos Speicherplatz reserviert werden kann. Für die hochauflösenden Grafikbetriebsarten (8,9,10), die jeweils 8138 Byte beanspruchen, ist es daher sinnvoller, die nachfolgende Maschinensprachroutine zum Laden von extern abgespeicherten Bildschirminhalten zu benutzen.

2.3.2 Laderoutine

Wenn Sie in einem Programm viele verschiedene Bildschirmseiten in hochauflösender Grafik verwenden wollen, können Sie die folgende Maschinenspracheroutine in Ihr Listing einarbeiten und beliebige GRAPHICS 8+16-Seiten von Diskette oder Kassette einlesen. Der Bildschirmaufbau geschieht dabei schneller als die normalen PLOT...DRAWTO-Befehle. Jede gespeicherte Bildschirmseite beansprucht ca. 62 Sektoren auf der Diskette (DOS 2).

```
0 REM BILD.LAD
10 GRAPHICS 8+16:DIM C$(7)
20 D=PEEK(560)+PEEK(561)*256
30 DN=PEEK(D+4)+PEEK(D+5)*256
40 OPEN #1,4,0,"D:BILD.DAT"
50 TRAP 120
60 A=848:POKE A+2,7:POKE A+5,INT(DN/256):POKE A+4,IN
T(DN-((PEEK(A+5)*256)))
70 B=7680:POKE A+9,INT(B/256):POKE A+8,INT(B-((PEEK(
A+9)*256)))
80 C$(1)=CHR$(104):C$(2)=CHR$(104):C$(3)=CHR$(104)
90 C$(4)=CHR$(170):C$(5)=CHR$(76):C$(6)=CHR$(86):C$(
7)=CHR$(228)
100 C=USR(ADR(C$),16)
110 RESTORE 300:FOR X=0 TO 60:READ Y:POKE 1536+X,Y:N
EXT X
120 CLOSE #1
130 GOTO 130
300 DATA 104,104,141,21,6,104,141,20,6,104,141,27,6,
104,141,26,6,160,193,173,255,255,136,240,35,141,255,
255,238
```

```
310 DATA 26,6,240,21,173,20,6,56,233,40,141,20,6,144
,4,24,76,19,6,206,21,6,76,19,6,238,27,6,76,33,6,96
```

Hier werden die Bildschirmdaten der Datei "BILD.DAT" eingelesen. Falls Sie mehrere Seiten gespeichert haben, arbeiten Sie am besten mit einer String-Variablen (z.B. IF SITUATION = 1 THEN DN$="D:BILD1":OPEN #1,4,0,DN$ usw.).

Um die Bilder abzuspeichern, können Sie folgendes Prinzip anwenden:

```
0 REM BILD.DAT
10 GRAPHICS 8+16
20 A=PEEK(88)+PEEK(89)*256
30 COLOR 1
40 PLOT 0,0:DRAWTO 319,0:DRAWTO 319,191:DRAWTO 0,191
:DRAWTO 0,0
50 PLOT 1,1:DRAWTO 318,1:DRAWTO 318,190:DRAWTO 1,190
:DRAWTO 1,1
60 PLOT 2,2:DRAWTO 317,189
70 PLOT 2,189:DRAWTO 317,2
80 PLOT 30,30:DRAWTO 290,30:DRAWTO 290,160:DRAWTO 30
,160:DRAWTO 30,30
90 PLOT 31,31:DRAWTO 289,31:DRAWTO 289,159:DRAWTO 31
,159:DRAWTO 31,31
100 OPEN #1,8,0,"D:BILD.DAT"
110 PUT #1,PEEK(A)
120 A=A+1:B=B+1:IF B=7680 THEN PUT #1,155:CLOSE #1:E
ND
130 GOTO 110
```

Das eigentliche Programm sind nur die Zeilen 20 und 100 bis 130. Mit den PLOT...DRAWTO-Befehlen (40 bis 90) wird ein Testbild erzeugt. Sie können hier die Grafik erstellen, die Sie später in Ihrem Programm aufrufen wollen.

2.3.3 Vertikales Scrollen

In manchen Programmen ist nicht der blitzschnelle Bildschirmwechsel gefragt, sondern das scheinbar endlose Rollen einer Grafik (z.B. einer Landschaft). Um dies zu erreichen, muß innerhalb der Display-Liste der Zeiger auf den Beginn des Bildschirmspeichers fortlaufend verändert werden. In GRAPHICS 0 (normale Betriebsart nach dem Einschalten) erhalten Sie folgende Display-Liste (800XL-Version):

```
112     Leerzeile
112     Leerzeile
112     Leerzeile
66      GR.0-Zeile (2) + Option der Anfangsadresse
        für den Bildschirmspeicher (64)
64      Low-Byte Bildschirmspeicher
152     High-Byte Bildschirmspeicher
2       23 verbleibende GR.0-Zeilen
2
2
2
2
2
2
2
2
2
2
2
2
2
2
2
2
2
2
2
2
2
2
65      Rücksprungbefehl zu:
32      Low-Byte Display-Liste
156     High-Byte Display-Liste
```

Mit dem nachfolgenden Programm können Sie die Display-Liste ausdrucken lassen:

```
0 REM DPL.GR0
10 GRAPHICS 0
20 A=PEEK(560)+PEEK(561)*256
30 FOR X=A TO A+32
40 LPRINT X;",";PEEK(X)
50 NEXT X
```

Schreiben Sie nun versuchsweise einmal: POKE A+4,PEEK(A+4)+40 (RETURN). Sie werden sehen, daß sich die Darstellung auf dem Bildschirm um eine Zeile nach oben schiebt. Das Low-Byte für die Adresse auf den Beginn des Bildschirmspeichers wird um 40 erhöht (da GRAPHICS 0 über 40 Spalten verfügt). Dies ist das Grundprinzip für vertikales Scrollen. In anderen Grafikbetriebsarten muß das Low-Byte um die Anzahl der jeweils verfügbaren Spalten erhöht werden (z.B. 80 in GRAPHICS 7). Mit folgendem Programm rollen 200 Zeilen auf dem Bildschirm vorüber.

```
0 REM VERTIKAL.SCR - HOCH
20 GRAPHICS 0
30 A=PEEK(560)+PEEK(561)*256
40 BL=PEEK(A+4):BH=PEEK(A+5)
50 FOR X=0 TO 200
60 BL=BL+40:IF BL>=256 THEN BL=BL-256:BH=BH+1:POKE A
+5,BH
70 POKE A+4,BL
80 FOR Z=0 TO 50:NEXT Z
90 NEXT X
100 GRAPHICS 0:LIST :END
```

Die Variablen BL und BH bedeuten "Bildschirmspeicher Low-Byte" und "Bildschirmspeicher High-Byte". In Zeile 40 wird BL bei jedem FOR...NEXT-Durchlauf um 40 erhöht. Sobald BL dabei 256 erreicht oder übersteigt, wird BH um Eins erhöht. Nach RUN (RETURN) sehen Sie auf dem Bildschirm "seltsame Zeichen". Es handelt sich um die Darstellung der Inhalte der Speicherstellen, die sich hinter dem normalen Bildschirmspeicher befinden. Sie können sich also mit dieser Methode den gesamten Speicher des Computers auf den Bildschirm holen.

Zum vertikalen Scrollen abwärts muß das Prinzip genau umgekehrt werden: BL=BL-40. Da sich beim augenblicklichen Programm in den Speicherstellen vor der Display-Liste nichts befindet (ungenutzter RAM-Bereich), werden im folgenden Beispielprogramm als Beginn des Bildschirmspeichers die Adressen 128 und 129 angegeben. Diese Adressen enthalten den sog. "Zeiger" (Low-Byte und High-Byte) auf das BASIC-Programm; d.h. ab

diesem Register ist das aktuelle Programm gespeichert. Sie werden hier jedoch nicht das normale Listing sehen, sondern das BASIC-Programm im sog. Token-Format. Das sind die maschineninternen Befehlsmarken.

```
0 REM VERTIKAL.SCR - RUNTER
20 GRAPHICS 0
30 A=PEEK(560)+PEEK(561)*256
40 BL=PEEK(128):BH=PEEK(129)
50 FOR X=0 TO 100
60 BL=BL-40:IF BL<0 THEN BL=BL+256:BH=BH-1:POKE A+5,
BH
70 POKE A+4,BL
80 FOR Z=0 TO 50:NEXT Z
90 NEXT X
100 GRAPHICS 0:LIST :END
```

2.3.4 Horizontales Scrollen

Zum horizontalen Scrollen wird die Adresse für den Bildschirmspeicher nicht zeilenweise, sondern spaltenweise weitergeschoben. Ändern Sie daher im vorhergehenden Programm Zeile 60:

```
60 BL=BL+1:IF BL>=256 THEN BL=BL-256:BH=BH+1:POKE A+
5,BH
```

Nach RUN (RETURN) rollt der Bildschirm um 200 Spalten nach links. Ändern Sie zum Scrollen nach rechts folgende Zeilen:

```
40 BL=PEEK(60000):BH=PEEK(60001)
60 BL=BL-1:IF BL<0 THEN BL=BL+256:BH=BH-1:POKE A+5,B
H
```

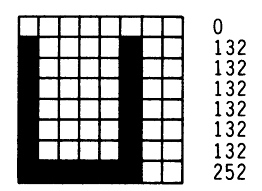
Diese Prinzipien zum horizontalen Scrollen sind für alle Grafikbetriebsarten identisch. Sie können sich also vom RAMTOP aus einen beliebigen Speicherbereich reservieren (so wie in den bisherigen Programmen oft gezeigt) und in die freien Speicherstellen Grafikpunkte oder Texte einPOKEn. Mit Hilfe der hier gezeigten Display-Manipulationen verändern Sie dann den Zeiger auf den Bildschirmspeicher, so daß die Grafiken auf dem Bildschirm vorüberrollen.

2.4 Player als Cursor

Die meisten Matrixdrucker bieten die Möglichkeit, selbstdefinierte Zeichen auszudrucken. Hierzu muß man, ähnlich wie bei den Zeichen des Computers, ein Punktmuster entwerfen. Bei den Druckermodellen "star radix 10" und "EPSON FX-80" steht dafür eine Matrix von 7 mal 11 Punkten zur Verfügung. Andere Modelle bieten 7 mal 9 Punkte o.ä.

So attraktiv, wie selbstdefinierte Druckerzeichen aussehen, so mühsam ist das Entwerfen auf Papier. Mit dem folgenden ENTWURF-Programm können Sie sich die Arbeit erleichtern: Auf dem Bildschirm erscheint ein Entwurfsraster und ein Player, der hier die Funktion eines Cursors übernimmt. Durch Joystick-Steuerung lassen sich Punkte setzen oder löschen. Auf Tastendruck werden die Dezimalwerte des Bitmusters ausgerechnet; das entworfene Zeichen kann dann gespeichert oder gedruckt werden. Verschiedene Sonderfunktionen erklärt das Programm während des Ablaufs selbsttätig.

Die vorliegende Version des ENTWURF-Programms ist auf den Drucker "star radix 10" abgestimmt. Falls Sie ein anderes Modell benutzen, müssen Sie nur wenige Zeilen umschreiben, um ein funktionsfähiges Utility zu erhalten (s. Erläuterungen zum Listing).




```
0 REM ENTWURF
10 POKE 106,PEEK(106)-8:GRAPHICS 0:DIM VH$(21),VR$(2
1):VP=48:HP=67:? "MOMENT BITTE"
20 POKE 752,1:DIM DC(11):DIM ZE$(1),X$(1),Y$(1):POKE
 709,204:POKE 710,146:ZD=10000
30 GOSUB 1370:? CHR$(125)
40 PRINT "  ┌─┬─┬─┬─┬─┬─┬─┬─┬─┬─┬─┐
50 PRINT "  │ │ │ │ │ │ │ │ │ │ │ │  1"
60 PRINT "  ├─┼─┼─┼─┼─┼─┼─┼─┼─┼─┼─┤
70 PRINT "  │ │ │ │ │ │ │ │ │ │ │ │  2"
80 PRINT "  ├─┼─┼─┼─┼─┼─┼─┼─┼─┼─┼─┤
90 PRINT "  │ │ │ │ │ │ │ │ │ │ │ │  4"
```

```
100 PRINT "  ├─┼─┼─┼─┼─┼─┼─┼─┼─┼─┼─┤
110 PRINT "  │ │ │ │ │ │ │ │ │ │ │ │  8"
120 PRINT "  ├─┼─┼─┼─┼─┼─┼─┼─┼─┼─┼─┤
130 PRINT "  │ │ │ │ │ │ │ │ │ │ │ │ 16"
140 PRINT "  ├─┼─┼─┼─┼─┼─┼─┼─┼─┼─┼─┤
150 PRINT "  │ │ │ │ │ │ │ │ │ │ │ │ 32"
160 PRINT "  ├─┼─┼─┼─┼─┼─┼─┼─┼─┼─┼─┤
170 PRINT "  │ │ │ │ │ │ │ │ │ │ │ │ 64"
180 PRINT "  └─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┘
190 POSITION 27,17:? "V=Vertikal"
200 POSITION 27,18:? "H=Horizont"
210 POSITION 27,19:? "N=Neu (leer)"
220 POSITION 2,23:? "Dezimalwerte = SELECT";
230 J=STICK(0):K=STRIG(0):O=PEEK(764)
240 IF J=7 AND HP<240 THEN HP=HP+8:SOUND 0,20,10,8
250 IF J=11 AND HP>8 THEN HP=HP-8:SOUND 0,20,10,8
260 IF J=14 AND VP>8 THEN VP=VP-1:X=USR(VH,P0+VP):GO
TO 230
270 IF J=13 AND VP<240 THEN POKE P0+VP,0:VP=VP+1:X=U
SR(VR,P0+VP):GOTO 230
280 IF K=0 THEN SOUND 0,8,8,10:GOSUB 1140:GOSUB 1330
300 IF O=16 THEN GOSUB 1140:GOSUB 740
310 IF O=80 THEN GOSUB 1140:GOSUB 750
320 IF O=57 AND HP=67 THEN GOSUB 1140:GOSUB 760
330 IF O=57 AND HP=75 THEN GOSUB 1140:GOSUB 770
340 IF O=121 AND HP=67 THEN GOSUB 1140:GOSUB 780
350 IF O=121 AND HP=75 THEN GOSUB 1140:GOSUB 790
360 IF O=35 THEN GOSUB 800
370 IF PEEK(53279)=5 THEN POSITION 2,23:? "Moment bi
tte.",,:GOSUB 400:REM 5=SELECT
380 POKE 53248,HP
390 FOR Z=0 TO 20:NEXT Z:SOUND 0,0,0,0:GOTO 230
400 FOR OFT=1 TO 11:DC(OFT)=0:NEXT OFT:SP=5:FOR OFT=
1 TO 11
410 ZAE=0:FOR L=2 TO 16 STEP 2:LOCATE SP,L,W
420 IF W=20 THEN DC(OFT)=DC(OFT)+(2^ZAE)
430 ZAE=ZAE+1:NEXT L
440 SP=SP+2:NEXT OFT
450 POSITION 2,20:FOR OFT=1 TO 11:? DC(OFT);",";:NEX
T OFT:? ,,,
460 POSITION 2,22:? "Weiterarbeiten = OPTION"
470 POSITION 2,23:? "Drucken = START  Speichern = S"
;
480 IF PEEK(53279)=6 THEN GOSUB 520
490 IF PEEK(53279)=3 THEN POSITION 2,22:? CHR$(157):
? "Dezimalwerte = SELECT",:RETURN
500 IF PEEK(764)=62 THEN GOSUB 820
```

```
510 GOTO 480
520 LPRINT CHR$(27);"A";CHR$(6);CHR$(27);"M";CHR$(9)
530 LPRINT CHR$(27);"B";CHR$(5);CHR$(27);"B";CHR$(2)
540 OPEN #1,8,0,"P:"
550 FOR PY=1 TO 16
560 FOR PX=4 TO 29:LOCATE PX,PY,W
570 IF W=1 THEN W=244
580 IF W=3 THEN W=247
590 IF W=4 THEN W=249
600 IF W=5 THEN W=242
610 IF W=17 THEN W=240
620 IF W=18 THEN W=241
630 IF W=19 THEN W=250
640 IF W=23 THEN W=243
650 IF W=24 THEN W=248
660 IF W=26 THEN W=246
670 IF W=124 THEN W=245
680 IF W=20 THEN W=174
690 PUT #1,W
700 NEXT PX:PUT #1,155:NEXT PY
710 LPRINT CHR$(27);"A";CHR$(12)
720 LPRINT "Dezimalwerte: ";DC(1);",";DC(2);",";DC(3
);","
730 LPRINT DC(4);",";DC(5);",";DC(6);",";DC(7);",";D
C(8);",";DC(9);",";DC(10);",";DC(11):CLOSE #1:RETURN

740 FOR L=PY TO 2 STEP -2:POSITION PX,L:? CHR$(20):S
OUND 0,L,10,8:NEXT L:POKE 764,255:RETURN
750 FOR L=PY TO 2 STEP -2:POSITION PX,L:? " ":SOUND
0,L,10,8:NEXT L:POKE 764,255:RETURN
760 FOR L=PX TO 25 STEP 4:POSITION L,PY:? CHR$(20):S
OUND 0,L,10,8:NEXT L:POKE 764,255:RETURN
770 FOR L=PX TO 23 STEP 4:POSITION L,PY:? CHR$(20):S
OUND 0,L,10,8:NEXT L:POKE 764,255:RETURN
780 FOR L=PX TO 25 STEP 4:POSITION L,PY:? " ":SOUND
0,L,10,8:NEXT L:POKE 764,255:RETURN
790 FOR L=PX TO 23 STEP 4:POSITION L,PY:? " ":SOUND
0,L,10,8:NEXT L:POKE 764,255:RETURN
820 POKE 764,255
830 POSITION 2,20:FOR V=1 TO 3:? CHR$(156);:NEXT V
840 POSITION 2,20:? "Welches Zeichen soll",:POSITION
 2,21:? "ersetzt werden ";:INPUT ZE$
850 POSITION 2,20:FOR V=1 TO 3:? CHR$(156);:NEXT V
860 POSITION 2,20:? "Welche Proportion":POSITION 2,2
1:? "(Werte 1 bis 11) ";:INPUT PR
870 ZD=ZD+ASC(ZE$):POSITION 2,21:? ZD;" DATA 27,42,1
,";ASC(ZE$);",";PR;",";
```

```
880 FOR OFT=1 TO 10:? DC(OFT);",";:NEXT OFT:? DC(11)
890 ? "CONT";:POSITION 2,19
900 POKE 842,13
910 END
920 POKE 842,12
930 POSITION 2,19:FOR V=1 TO 3:? CHR$(156);:NEXT V:P
OSITION 2,0:? CHR$(157)
940 POSITION 2,20:? "Drucken = D":? "Weiterarbeiten
= OPTION"
950 IF PEEK(764)=58 THEN GOSUB 980
960 IF PEEK(53279)=3 THEN POSITION 2,19:FOR V=1 TO 4
:? CHR$(157):NEXT V:RUN
970 GOTO 950
980 LPRINT CHR$(27);"@";CHR$(27);"B";CHR$(5);CHR$(27
);CHR$(42);CHR$(0)
990 POKE 764,255:OPEN #1,8,0,"P:":TRAP 1010
1000 POSITION 2,20:? "Moment...Ich lese Daten.":? CH
R$(156):READ D:PUT #1,D:GOTO 1000
1010 CLOSE #1:POSITION 2,20:? "Alles drucken (J/N)
        ":? CHR$(156)
1020 IF PEEK(764)=1 THEN POKE 764,255:GOTO 1100
1030 IF PEEK(764)=35 THEN POKE 764,255:GOTO 1050
1040 GOTO 1020
1050 POSITION 2,20:? "Welches Zeichen soll gedruckt"
:? "werden";:INPUT ZE$
1060 LPRINT CHR$(27);CHR$(36);CHR$(1)
1070 LPRINT ZE$
1080 LPRINT CHR$(27);CHR$(36);CHR$(0)
1090 RUN
1100 POSITION 2,20:? "Zeichen von",,:INPUT X$:? "bis
          ",:INPUT Y$
1110 LPRINT CHR$(27);CHR$(36);CHR$(1)
1120 FOR V=ASC(X$) TO ASC(Y$):LPRINT CHR$(V):NEXT V
1130 LPRINT CHR$(27);CHR$(36);CHR$(0):RUN
1140 REM
1150 IF HP=67 THEN PX=5
1160 IF HP=75 THEN PX=7
1170 IF HP=83 THEN PX=9
1180 IF HP=91 THEN PX=11
1190 IF HP=99 THEN PX=13
1200 IF HP=107 THEN PX=15
1210 IF HP=115 THEN PX=17
1220 IF HP=123 THEN PX=19
1230 IF HP=131 THEN PX=21
1240 IF HP=139 THEN PX=23
1250 IF HP=147 THEN PX=25
1260 IF VP<50 THEN PY=2:RETURN
```

```
1270 IF VP<66 THEN PY=4:RETURN
1280 IF VP<82 THEN PY=6:RETURN
1290 IF VP<98 THEN PY=8:RETURN
1300 IF VP<114 THEN PY=10:RETURN
1310 IF VP<130 THEN PY=12:RETURN
1320 IF VP<146 THEN PY=14:RETURN
1330 LOCATE PX,PY,PZ
1340 IF PZ=20 THEN POSITION PX,PY:? " "
1350 IF PZ=32 THEN POSITION PX,PY:? CHR$(20)
1360 RETURN
1370 REM PM-TABELLE
1380 POKE 54279,PEEK(106):PMT=PEEK(106)*256
1390 FOR L=PMT+1024 TO PMT+1279:POKE L,0:NEXT L:P0=P
MT+1024
1400 POKE 559,58:POKE 53277,2:POKE 704,206:POKE 623,
1
1410 FOR L=0 TO 7:READ D:POKE P0+L+VP,D:NEXT L
1420 DATA 0,132,132,132,132,132,132,252
1430 VH=ADR(VH$):VR=ADR(VR$)
1440 FOR L=VH TO VH+20:READ D:POKE L,D:NEXT L
1450 FOR L=VR TO VR+20:READ D:POKE L,D:NEXT L:RETURN

1460 DATA 104,104,133,204,104,133,203,160,1,177
1470 DATA 203,136,145,203,200,200,192,17,208,245,96
1480 DATA 104,104,133,204,104,133,203,160,16,177
1490 DATA 203,200,145,203,136,136,192,255,208,245,96
```

10: Für Player-Missile-Grafik mit einzeiliger Auflösung werden acht "Seiten" vom RAMTOP aus reserviert. VP und HP sind die Variablen für die Vertikal- und Horizontalposition des Players.
20: Mit POKE 752,1 wird der Textcursor unsichtbar (normal = 0). Die indizierte Variable DC(x) wird für die Dezimalwerte der definierten Zeichen benötigt.
30: Im Unterprogramm ab Zeile 1370 wird die Player-Missile-Tabelle aufgebaut.
40 bis 180: Die PRINT-Anweisungen für das Entwurfsraster. Die Zeichen erreichen Sie durch Drücken von CONTROL+Q, CONTROL+W usw. (s. Anleitungsbuch zum ATARI). Falls Sie einen Matrixdrucker benutzen, der nicht über 7 mal 11 Nadelpunkte verfügt, müssen Sie das Raster hier entsprechend ändern.
190: Durch Drücken von V können Sie eine Bitreihe vertikal mit Punkten füllen. Drücken Sie SHIFT+V, wird die Reihe wieder gelöscht.
200: Wenn Sie H drücken, füllt sich eine Entwurfsreihe horizontal mit Punkten, jedoch nur, wenn sich der Player in der ersten oder zweiten Spalte befindet. Durch Drücken von SHIFT+H können Sie die Reihe horizontal wieder löschen.
210: Mit der Taste N werden alle bisher gesetzten Punkte gelöscht; Sie erhalten also ein neues Entwurfsfeld.
220: Wenn Sie die Konsolentaste SELECT drücken, werden die Dezimalwerte des entworfenen Zeichens auf dem Bildschirm ausgegeben. Sie können danach das Zeichen speichern oder zur Eingabe zurückkehren.
230: Die Abfrage des Joysticks und des Tastaturregisters 764.
240 bis 270: Je nach Bewegung des Steuerknüppels bewegt sich der Player auf dem Bildschirm. Dazu ertönt ein akustisches Signal.
280: Wenn der Knopf gedrückt wird, lokalisiert der Rechner im Unterprogramm ab Zeile 1140, an welcher Stelle sich der Player befindet und ordnet den Variablen PX und PY die damit korrespondierende Bildschirmposition zu. An dieser Position wird dann entweder ein Punkt gesetzt oder gelöscht, je nachdem, was sich dort befindet.
300 bis 370: Die Tastaturfunktionen für vertikales und horizontales Füllen oder Löschen der anvisierten Reihe bzw. Spalte. Die Werte, die je nach Tastendruck in Adresse 764 registriert werden, entnehmen Sie der Tabelle im Anhang.
380: Die aktuelle Horizontalposition des Players wird gePOKEt.
400 bis 440: In dieses Unterprogramm springt der Rechner, sobald Sie SELECT drücken. Durch die FOR...NEXT-Schleife werden zunächst die Bildschirmpunkte des Entwurfsfeldes lokalisiert und anschließend die Dezimalwerte des Zeichens ausgerechnet.

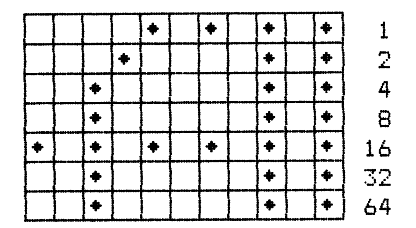
Im Gegensatz zu den Bitmustern für Player oder ATARI-Zeichen sind die Bits bei Druckern vertikal angeordnet. Die Dezimalwerte werden also nicht durch Zusammenzählen von links nach rechts, sondern von oben nach unten ausgerechnet. Allerdings gibt es hier auch Unterschiede: Bei einigen Druckermodellen ist das höherwertige Bit oben, bei anderen unten. Schauen Sie deshalb in der Anleitung zu Ihrem Drucker nach, falls Sie keinen "star radix 10" benutzen; ggf. müssen Sie die FOR...NEXT-Schleife von 2 TO 16 STEP 2 ändern in 16 TO 2 STEP -2. Der Rechenvorgang in Zeile 420 bleibt erhalten.
450: Die Dezimalwerte werden auf dem Bildschirm ausgegeben.
460 bis 510: Sie können nun entscheiden, ob Sie weiterarbeiten, das Zeichen speichern oder den Entwurf ausdrucken wollen.
520 bis 730: In dieses Unterprogramm springt der Rechner, um den Entwurf auszudrucken. Die Druckeranweisungen für den "EPSON FX-80" entnehmen Sie den Listing-Ergänzungen am Ende dieses Kommentars.
740 bis 810: Hier werden die Sonderfunktionen wie "Reihe vertikal füllen" ausgeführt. Der Rechner lokalisiert dabei jeweils die Position des Players und setzt oder löscht von da an die Punkte.
820 bis 970: Wenn Sie die Option "Zeichen speichern" wählen, wird in diesem Unterprogramm gefragt, um welches Zeichen es sich handelt. Danach müssen Sie die Porportion des Zeichens eingeben (falls Sie keine Proportionalschrift verwenden, geben Sie hier immer die Zahl 11 ein). Durch eine automatische RETURN-Funktion wird anschließend eine DATA-Zeile mit den Dezimalwerten des Zeichens generiert.
980 bis 1130: Sobald ein Entwurf gespeichert ist, können Sie dieses Zeichen oder auch einen kompletten Zeichensatz ausdrucken lassen. Der Aufruf eines selbstdefinierten Zeichensatzes (Download Character-Set) ist bei jedem Drucker verschieden. Schauen Sie daher in Ihre Bedienungsanleitung und ersetzen Sie die CHR$-Codes in den Zeilen 980, 1060 und 1080 durch die für Sie zutreffenden Werte.
1140 bis 1320: Sobald während des Entwerfens der Feuerknopf gedrückt wird, springt der Rechner in dieses Unterprogramm und lokalisiert, an welcher Bildschirmposition der Player steht. Für die Vertikalposition wurden dabei Toleranzen von -2 bis +2 gegeben; d.h. der Punkt wird auch gesetzt, wenn Sie die betreffende Bildschimposition nur "ungefähr" ansteuern.
1330 bis 1360: Falls sich an der angesteuerten Position ein Punkt befindet, wird er gelöscht; falls sich eine Leerstelle dort befindet, wird ein Punkt gesetzt.
1370 bis 1490: Die Player-Missile-Tabelle mit dem Maschinenspracheprogramm zur Steuerung der Vertikalbewegungen des

Players, so wie im ersten Kapitel des Buches erklärt.

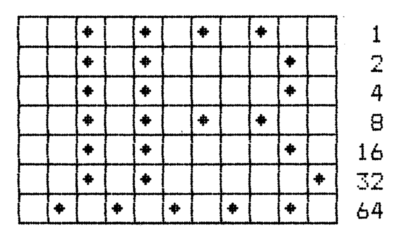
Listing-Ergänzung zum Ausdruck des Entwurfsfeldes mit "EPSON FX-80" (die Zeilen 580 bis 680 müssen gelöscht werden):

```
520 LPRINT :LPRINT CHR$(27);"A";CHR$(6);CHR$(27);"1"
;CHR$(9)
530 LPRINT CHR$(27);CHR$(15)
540 OPEN #1,8,0,"P:"
550 FOR PY=2 TO 14 STEP 2
560 FOR PX=5 TO 25 STEP 2:LOCATE PX,PY,W
570 IF W=20 THEN W=42
690 PUT #1,W
700 NEXT PX:PUT #1,155:NEXT PY
710 LPRINT CHR$(27);"A";CHR$(12);CHR$(18)
```

Beispiel für den Ausdruck eines Entwurfsfeldes mit "star radix-10"

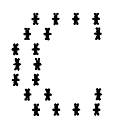


Dezimalwerte: 16,0,124,
2,17,0,17,0,127,0,127



Dezimalwerte: 0,64,63,
64,63,64,9,64,9,86,32

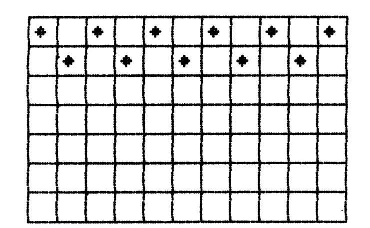
Beispiel für den Ausdruck eines Entwurfsfeldes mit "EPSON FX-80"



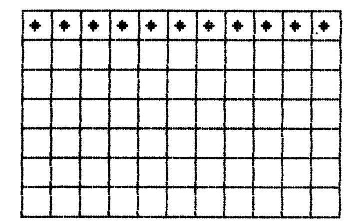
Dezimalwerte: 0,28,34,
93,34,65,0,65,0,99,0

Dezimalwerte: 1,0,127,
0,127,0,65,0,65,34,28

Beim Entwerfen müssen Sie eine Besonderheit beachten: Direkt nebeneinanderliegende Punkte dürfen nicht gesetzt werden. Dies würde vom Drucker ignoriert. Die Darstellung auf der folgenden Seite verdeutlicht, was gemeint ist:



erlaubt



nicht erlaubt

Beispiel für einen selbstdefinierten Zeichensatz.

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

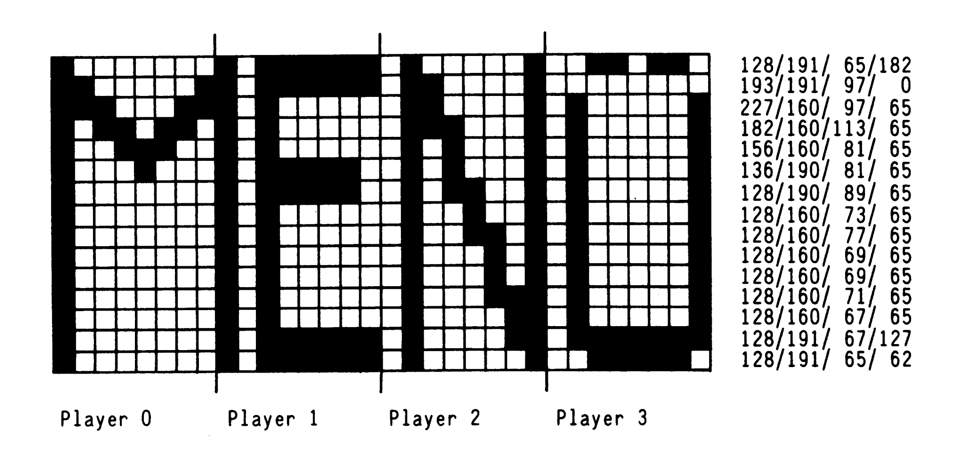
Und hier die dazugehörenden DATA-Zeilen:

```
10064 DATA 27,42,1,64,11,0,28,34,73,20,65,28,65,18,7
6,0
10065 DATA 27,42,1,65,11,16,0,124,2,17,0,17,0,127,0,
127
10066 DATA 27,42,1,66,11,0,64,63,64,63,64,9,64,9,86,
32
10067 DATA 27,42,1,67,11,0,0,28,34,93,34,65,0,65,0,9
9
10068 DATA 27,42,1,68,11,1,0,127,0,127,0,65,0,65,34,
28
10069 DATA 27,42,1,69,11,65,0,127,0,127,0,73,20,65,0
,99
10070 DATA 27,42,1,70,11,1,0,127,0,127,0,9,20,1,0,3
10071 DATA 27,42,1,71,11,0,0,28,34,93,34,65,0,73,0,1
23
10072 DATA 27,42,1,72,11,8,0,127,0,127,0,8,0,8,0,127
10073 DATA 27,42,1,73,11,0,0,0,127,0,127,0,1,0,0,0
10074 DATA 27,42,1,74,11,0,24,32,64,0,64,63,64,63,0,
1
10075 DATA 27,42,1,75,11,1,0,127,0,127,0,28,34,65,0,
65
10076 DATA 27,42,1,76,11,1,0,127,0,127,0,64,0,64,0,9
6
10077 DATA 27,42,1,77,11,127,0,127,0,2,12,0,12,66,1,
127
10078 DATA 27,42,1,78,11,1,0,127,0,127,0,2,4,24,0,12
7
10079 DATA 27,42,1,79,11,0,28,34,93,34,65,0,65,0,65,
62
```

```
10080 DATA 27,42,1,80,11,1,0,127,0,127,0,17,0,17,10,
4
10081 DATA 27,42,1,81,11,0,28,34,93,34,65,16,65,112,
1,94
10082 DATA 27,42,1,82,11,1,0,127,0,127,0,17,0,113,10
,68
10083 DATA 27,42,1,83,11,0,0,70,9,70,9,68,9,84,41,16
10084 DATA 27,42,1,84,11,2,1,0,127,0,127,0,3,0,3,0
10085 DATA 27,42,1,85,11,1,0,63,0,127,0,64,0,64,0,63
10086 DATA 27,42,1,86,11,1,0,31,32,31,96,0,64,32,31,
0
10087 DATA 27,42,1,87,11,127,0,127,0,32,16,8,16,33,6
4,63
10088 DATA 27,42,1,88,11,1,64,33,18,5,10,20,40,82,33
,64
10089 DATA 27,42,1,89,11,0,3,4,3,124,0,124,3,4,3,0
10090 DATA 27,42,1,90,11,0,0,99,16,105,20,75,4,99,0,
0
```

2.5 Player als Textzeichen

Über zusammengefügte Player, die ein Spielobjekt (Raumschiff, Fahrzeug etc.) ergeben, wurde im Abschnitt 1.6.4 bereits geschrieben. Diese "Technik" kann jedoch auch für Schriftzeichen oder Grafiksymbole genutz werden. Sie erhalten damit mehrfarbige Zeichen in beliebiger Höhe. Durch die Möglichkeit zur doppelten oder vierfachen Breite können Sie weitere Darstellungsformen auf dem Bildschirm nutzen. Als Einsatzbereiche bieten sich z.B. Grafikbetriebsarten an, die über keinen Textmodus verfügen (GRAPHICS 9 bis 11).



Das folgende Anwendungsbeispiel läßt das Wort "MENÜ" in normaler, doppelter und vierfacher Breite erscheinen:

```
0 REM PLAYER ALS TEXTZEICHEN
10 POKE 106,PEEK(106)-8
20 GRAPHICS 3
30 ? "MOMENT BITTE"
40 POKE 54279,PEEK(106)
50 PMT=PEEK(106)*256
60 FOR L=PMT+1024 TO PMT+2048
70 POKE L,0
80 NEXT L
90 P0=PMT+1024
100 P1=P0+256
110 P2=P1+256
120 P3=P2+256
130 FOR L=0 TO 14:READ D:POKE P0+L+50,D:NEXT L
140 FOR L=0 TO 14:READ D:POKE P1+L+50,D:NEXT L
150 FOR L=0 TO 14:READ D:POKE P2+L+50,D:NEXT L
160 FOR L=0 TO 14:READ D:POKE P3+L+50,D:NEXT L
170 POKE 559,62
180 POKE 53277,2
190 POKE 704,110
200 POKE 705,110
210 POKE 706,110
220 POKE 707,110
230 POKE 623,1
240 ? CHR$(125)
250 POKE 53248,64
260 POKE 53249,72
270 POKE 53250,80
280 POKE 53251,88
290 FOR L=0 TO 500:NEXT L
300 REM DOPPELTE BREITE
310 POKE 53248,64
320 POKE 53249,80
330 POKE 53250,96
340 POKE 53251,112
350 POKE 53256,1
360 POKE 53257,1
370 POKE 53258,1
380 POKE 53259,1
390 FOR L=0 TO 500:NEXT L
400 REM VIERFACHE BREITE
410 POKE 53248,64
420 POKE 53249,96
```

```
430 POKE 53250,128
440 POKE 53251,160
450 POKE 53256,3
460 POKE 53257,3
470 POKE 53258,3
480 POKE 53259,3
490 FOR L=0 TO 500:NEXT L
500 POKE 53256,0
510 POKE 53257,0
520 POKE 53258,0
530 POKE 53259,0
540 GOTO 250
550 DATA 128,193,227,182,156,136,128,128,128,128,128
,128,128,128,128
560 DATA 191,191,160,160,160,190,190,160,160,160,160
,160,160,191,191
570 DATA 65,97,97,113,81,81,89,73,77,69,69,71,67,67,
65
580 DATA 54,0,65,65,65,65,65,65,65,65,65,65,65,127,6
2
```

10: Vom RAMTOP aus werden acht "Seiten" für Player-Missile-Grafik mit einzeiliger Auflösung reserviert.
40 bis 120: Aufbau der PM-Tabelle und Festlegung der Anfangsadressen für die einzelnen Player innerhalb der Tabelle.
140 bis 160: Die Dezimalwerte für die Textzeichen werden in die Player-Adressen gePOKEt.
190 bis 220: Festlegung der Farbwerte für die Player.
250 bis 280: Bei normaler Breite liegen die Horizontalpositionen der Player jeweils 8 Spalten auseinander.
310 bis 340: Zur Darstellung der Zeichen in doppelter Breite müssen die Horizontalpositionen der Player 16 Spalten auseinander liegen.
350 bis 380: Die Register zur Festlegung der Breite werden auf doppelte Breite geschaltet.
410 bis 440: Sollen die Textzeichen in vierfacher Breite erscheinen (sie nehmen dann fast die gesamte Bildschirmbreite ein), müssen die Horizontalpositionen 32 Spalten auseinander liegen.
500 bis 530: Die Register zur Festlegung der Breite werden wieder auf Null gesetzt (normale Breite).
540: Rücksprung zur Zeile 250. Sie können das Programm durch Drücken von RESET unterbrechen.

Nun ist es allerdings ziemlich mühsam, beim Entwerfen solcher Zeichen jedesmal die Dezimalwerte der vier Player auszurech-

nen. Falls Sie also Player öfters für Text oder Grafiksymbole einsetzen wollen, wird das folgende Listing für Sie nützlich sein: In der hochauflösenden Grafikbetriebsart 7 erscheint ein Raster von 32 mal 15 Feldern, (für vier Player mit je 15 Zeilen Höhe). Mit Hilfe eines Joysticks (Port 1) können Sie alle gewünschten Textzeichen oder Symbole entwerfen. Auf Tastendruck erscheinen dann die Player mit dem entworfenen Muster, so daß Sie ständig die optische Wirkung in Originalgröße überprüfen können. Durch weiteren Tastendruck werden die Dezimalwerte für die Player ausgegeben und das Programm beendet.

```
0 REM PLAYER ALS ZEICHEN - ENTWURFSRASTER
10 POKE 106,PEEK(106)-8:GRAPHICS 7
20 DIM D0(14),D1(14),D2(14),D3(14):HX=11:HY=47:VX=8:
VY=1
30 GOSUB 660
40 SETCOLOR 4,0,4:SETCOLOR 0,0,12:SETCOLOR 1,8,12:SE
TCOLOR 2,2,4
50 COLOR 1:POKE 752,1
60 FOR L=0 TO 45 STEP 3:PLOT 10,L:DRAWTO 106,L:NEXT
L
70 FOR L=10 TO 106 STEP 3:PLOT L,0:DRAWTO L,45:NEXT
L
80 ? "OPTION = Player ein/aus"
90 ? "SELECT = Dezimalwerte"
100 ? "START  = Neu"
110 J=STICK(0):K=STRIG(0):O=PEEK(53279)
120 IF J=7 AND HX<103 THEN GOSUB 220:HX=HX+3
130 IF J=11 AND HX>11 THEN GOSUB 220:HX=HX-3
140 IF J=13 AND VY<42 THEN GOSUB 220:VY=VY+3
150 IF J=14 AND VY>2 THEN GOSUB 220:VY=VY-3
160 IF K=0 THEN GOSUB 230
170 IF O=3 THEN GOSUB 760:GOSUB 280
180 IF O=5 THEN ? CHR$(125);"MOMENT BITTE";:GOSUB 76
0:GOSUB 380:GOTO 560
190 IF O=6 THEN GRAPHICS 7:GOTO 40
200 COLOR 2:PLOT HX,HY:PLOT VX,VY
210 FOR L=0 TO 20:NEXT L:SOUND 0,0,0,0:GOTO 110
220 COLOR 0:PLOT HX,HY:PLOT VX,VY:SOUND 0,10,12,6:RE
TURN
230 LOCATE HX,VY,A:SOUND 0,10,8,10
240 IF A=0 THEN C=1
250 IF A=1 THEN C=0
260 COLOR C:PLOT HX,VY:PLOT HX+1,VY:PLOT HX,VY+1:PLO
T HX+1,VY+1
270 RETURN
```

```
280 REM PLAYER ANSEHEN
290 SOUND 0,122,10,4:SOUND 1,97,10,4:IF H=100 THEN H
=0:GOTO 360
300 GOSUB 380
310 FOR L=0 TO 14:POKE P0+L+150,D0(L):NEXT L
320 FOR L=0 TO 14:POKE P1+L+150,D1(L):NEXT L
330 FOR L=0 TO 14:POKE P2+L+150,D2(L):NEXT L
340 FOR L=0 TO 14:POKE P3+L+150,D3(L):NEXT L
350 IF H=0 THEN H=100
360 POKE 53248,H:POKE 53249,H+8:POKE 53250,H+16:POKE
 53251,H+24
370 SOUND 0,0,0,0:SOUND 1,0,0,0:RETURN
380 REM DEZIMALWERTE
390 X=0:FOR M=1 TO 43 STEP 3:Z=7:FOR L=11 TO 32 STEP
 3
400 LOCATE L,M,A
410 IF A=1 THEN D0(X)=D0(X)+(2^Z)
420 Z=Z-1:NEXT L:X=X+1:NEXT M
430 X=0:FOR M=1 TO 43 STEP 3:Z=7:FOR L=35 TO 56 STEP
 3
440 LOCATE L,M,A
450 IF A=1 THEN D1(X)=D1(X)+(2^Z)
460 Z=Z-1:NEXT L:X=X+1:NEXT M
470 X=0:FOR M=1 TO 43 STEP 3:Z=7:FOR L=59 TO 80 STEP
 3
480 LOCATE L,M,A
490 IF A=1 THEN D2(X)=D2(X)+(2^Z)
500 Z=Z-1:NEXT L:X=X+1:NEXT M
510 X=0:FOR M=1 TO 43 STEP 3:Z=7:FOR L=83 TO 104 STE
P 3
520 LOCATE L,M,A
530 IF A=1 THEN D3(X)=D3(X)+(2^Z)
540 Z=Z-1:NEXT L:X=X+1:NEXT M
550 RETURN
560 GRAPHICS 0
570 ? :? :? "PLAYER 0 = ";
580 FOR L=0 TO 14:? D0(L);",";:NEXT L
590 ? :? :? "PLAYER 1 = ";
600 FOR L=0 TO 14:? D1(L);",";:NEXT L
610 ? :? :? "PLAYER 2 = ";
620 FOR L=0 TO 14:? D2(L);",";:NEXT L
630 ? :? :? "PLAYER 3 = ";
640 FOR L=0 TO 14:? D3(L);",";:NEXT L
650 END
660 REM PM-TABELLE
670 POKE 54279,PEEK(106):? "MOMENT BITTE"
680 PMT=PEEK(106)*256
```

```
690 FOR L=PMT+1024 TO PMT+2048
700 POKE L,0:NEXT L
710 P0=PMT+1024:P1=P0+256:P2=P1+256:P3=P2+256
720 POKE 559,58
730 POKE 53277,2
740 POKE 704,254:POKE 705,254:POKE 706,254:POKE 707,
254
750 ? CHR$(125)
760 FOR L=0 TO 14:D0(L)=0:NEXT L
770 FOR L=0 TO 14:D1(L)=0:NEXT L
780 FOR L=0 TO 14:D2(L)=0:NEXT L
790 FOR L=0 TO 14:D3(L)=0:NEXT L
800 RETURN
```

10: Für die Player, die während des Programms eingeblendet werden können, wird einzeilige Auflösung gewählt. Falls Sie Ihre Zeichen in der gröberen Zweizeilen-Auflösung darstellen wollen, genügen vier reservierte "Seiten". Außerdem müssen Sie die Player-Missile-Tabelle ändern (s. Erläuterungen im 1. Kapitel für zweizeilige Auflösung).
20: Für die Dezimalwerte der Player werden vier Variablen mit je 15 Elementen DIMensioniert.
40: Die Farben für Hintergrund, Textfenster, Entwurfsraster und die beiden Randcursor.
80 bis 100: Schreiben Sie kursiv erscheinende Wörter zum besseren Erkennen invertiert (ATARI-Taste).
110: Abfrage des Joysticks und der Konsolentasten (OPTION usw.).
120 bis 150: Je nach Joystick-Bewegung verändern sich die Positionen des horizontalen und vertikalen Cursors.
160: Sobald der Feuerknopf gedrückt wird, springt der Rechner ins Unterprogramm 230. Befindet sich an der angesteuerten Bildschirmposition ein Punkt, wird er gelöscht; im anderen Fall erscheint ein Punkt.
170: Wenn Sie OPTION drücken, werden im Unterprogramm ab Zeile 280 die aktuellen Dezimalwerte in die Player-Register gePOKEt. Danach erscheinen die Player auf dem Bildschirm. Wenn Sie erneut OPTION drücken, verschwinden die Player wieder und Sie können mit Ihrer Arbeit fortfahren.
180: Durch Drücken von SELECT wird das Programm beendet. Auf dem Bildschirm erscheinen die Dezimalwerte der einzelnen Player.
190: Mit der Taste START wird das gesamte Entwurfsfeld gelöscht, so daß Sie neue Zeichen entwerfen können.
200: Die Randcursor erscheinen an den aktuellen Positionen.
220: Bevor ein Cursor an neuer Position erscheinen kann, muß

die Darstellung an der alten Position gelöscht werden. Dies wird hier in Zeile 220 vorgenommen.
230 bis 270: Setzen oder Löschen eines Grafikpunktes bei Betätigung des Feuerknopfes.
290: Da das Einlesen der Daten für die Player einige Sekunden dauert, ertönt zunächst ein Akkord, der deutlich macht, daß der Computer arbeitet.
310 bis 340: Die im Unterprogramm ab Zeile 380 ausgerechneten Dezimalwerte werden in die Player-Register gePOKEt.
360: Im Abstand von acht Spalten erscheinen die vier Player nebeneinander (normale Breite). Falls Sie in Ihren Programmen Textzeichen lieber in doppelter oder vierfacher Breite darstellen wollen, müssen Sie die Player im Abstand von 16 bzw. 32 Spalten auf den Bildschirm bringen.
390 bis 550: Hier werden die Dezimalwerte für die Player ausgerechnet.
560 bis 650: Sobald Sie die Taste SELECT drücken, erscheinen die aktuellen Dezimalwerte für die Player auf dem Bildschirm, und das Programm wird beendet. Sie können die Zahlen durch eine LPRINT-Anweisung natürlich auch vom Drucker ausgeben lassen.
660 bis 740: Aufbau der PM-Tabelle, so wie im 1. Kapitel des Buches bereits erklärt.
760 bis 790: Zu Beginn des Programms und bei Änderungen der entworfenen Zeichen werden die indizierten Variablen auf Null gesetzt, da sonst möglicherweise die alten Werte die Bildschirmausgabe durcheinander bringen.

3. Kapitel

Utilities

3 Utilities

Falls Sie durch dieses Buch angeregt wurden, eigene Player-Objekte zu entwerfen, werden Sie festgestellt haben, daß der Entwurf und das Ausrechnen der Dezimalwerte relativ aufwendig ist: Man muß ein Raster zeichnen, ggf. mit dem Radiergummi Fehlentwürfe löschen usw. Was liegt da näher, als ein Programm zu entwickeln, mit dem sich Player besonders komfortabel kreieren lassen? - In diesem Kapitel wird Ihnen solch ein Utility geboten.

Ein anderes Problem, das immer wieder auftritt, ist die Farbwahl für Player und Bildschirmgrafik. Immerhin bieten die ATARI-Computer 256 verschiedene Farben. Wenn Sie z.B. vier Farbregister für den Hintergrund und weitere vier für die Player wählen wollen, können Sie nur durch mühsames Probieren die Wirkung der einzelnen Farbkombinationen überprüfen. Als zweites Utility-Programm finden Sie daher ein "Testbild", bei dem Sie durch Drücken verschiedener Tasten sämtliche Farben aufrufen können. Auf weiteren Knopfdruck werden Ihnen dann die Dezimalwerte bzw. SETCOLOR-Angaben gezeigt.

3.1 Player-Missile-Editor

Mit dem folgenden Player-Missile-Editor können Sie bequem mit einem Joystick Ihr gewünschtes Bitmuster entwerfen, beliebig verändern oder ganz löschen. Viele Sonderfunktionen, wie das Ausfüllen einer ganzen Reihe oder das Invertieren des entworfenen Objektes, bringen weitere Arbeitserleichterungen.

Bedienung des Player-Missile-Editors

Joystick an Port 1:
Steuerung der Randcursor am Entwurfsraster
durch Rechts/Linksbewegung

Einzelnen Punkt im Entwurfsraster setzen:
Feuerknopf drücken

Gesetzten Punkt wieder löschen:
Feuerknopf drücken

F = gesamtes Raster füllen
L = gesamtes Raster löschen

I = entworfenes Objekt invertieren
J = Invertierung rückgängig machen
H = eine gesamte Bitreihe horizontal setzen
G = eine gesamte Bitreihe horizontal löschen
V = eine gesamte Bitreihe vertikal setzen
B = eine gesamte Bitreihe vertikal löschen

OPTION = Dezimalwerte für ein Objekt speichern
SELECT = Entwurf als Datei sichern

1. Speichern Sie das Programm vor dem ersten Durchlauf im LIST- oder SAVE-Format auf Diskette oder Kassette. Die Dezimalwerte für die entworfenen Objekte werden später im Speicher des Rechners verwaltet und würden bei Sicherung des Programms unnötig Sektoren belegen.

2. Auf dem Bildschirm erscheint ein Entwurfsraster von 10*8 Feldern und zwei Koordinatenkreuze, die durch den Joystick bewegt werden. Wenn Sie an einer bestimmten Stelle einen Punkt setzen wollen, drücken Sie den Feuerknopf. Auf die gleiche Weise können Sie einen Punkt auch wieder löschen.

3. Alle Sonderfunktionen sind neben dem Entwurfsraster aufgelistet. In der Praxis haben sie sich als besonders nützlich herausgestellt. Wenn Sie z.B. sieben Punkte in einer Reihe setzen wollen, müssen Sie nicht siebenmal den Feuerknopf drücken, sondern füllen die betreffende Reihe durch Drücken von H und löschen den nichtgewünschten achten Punkt.

4. Mit OPTION sichern Sie die Daten eines entworfenen Objektes.

5. Mit SELECT werden die Daten aller von Ihnen entworfenen Player automatisch als Datei auf Diskette gespeichert. Falls Sie einen Datenrecorder benutzen, müssen Sie lediglich einen einzelnen Befehl ändern (Zeile 2030).

```
0 REM PLAYER-MISSILE EDITOR
10 DIM DW(9),Z$(12),N$(14):ZN=10000
20 GRAPHICS 0:POKE 709,204:POKE 710,146:POKE 752,1:?
 CHR$(125)
30 PX=2:PY=1:POSITION PX,PY
40 FOR X=0 TO 16:READ D:? CHR$(D);:NEXT X
50 PY=PY+1:POSITION PX,PY
60 FOR Y=1 TO 9
70 FOR X=0 TO 16:READ D:? CHR$(D);:NEXT X
```

```
80 PY=PY+1:POSITION PX,PY
90 FOR X=0 TO 16:READ D:? CHR$(D);:NEXT X
100 RESTORE 6010:PY=PY+1:POSITION PX,PY:NEXT Y
110 POSITION PX,PY
120 FOR X=0 TO 16:READ D:? CHR$(D);:NEXT X
130 RESTORE 6040:PY=PY+1:POSITION PX,PY
140 FOR X=0 TO 16:READ D:? CHR$(D);:NEXT X
150 HX=1:HY=20:VX=3:VY=22
160 POSITION HX,HY:? CHR$(19)
170 POSITION VX,VY:? CHR$(19)
180 POSITION 22,1:? "F Raster fuellen";
190 POSITION 22,2:? "L Raster loeschen";
200 POSITION 22,4:? "I Invers ein";
210 POSITION 22,5:? "J Invers aus";
220 POSITION 22,7:? "H horizontal voll";
230 POSITION 22,8:? "G horizontal leer";
240 POSITION 22,10:? "V vertikal voll";
250 POSITION 22,11:? "B vertikal leer";
260 POSITION 22,13:? "OPTION Player";
270 POSITION 22,14:? "       sichern";
280 POSITION 22,16:? "SELECT Datei";
290 POSITION 22,17:? "       sichern";
300 T=PEEK(764):J=STICK(0):K=STRIG(0)
310 IF T=56 THEN GOSUB 1000
320 IF T=0 THEN GOSUB 1100
330 IF T=13 OR T=1 THEN GOSUB 1200
340 IF T=57 THEN GOSUB 1500
350 IF T=61 THEN GOSUB 1600
360 IF T=16 THEN GOSUB 1700
370 IF T=21 THEN GOSUB 1800
390 IF PEEK(53279)=3 THEN 1300
400 IF PEEK(53279)=5 THEN 2000
410 IF J=14 AND HY>2 THEN POSITION HX,HY:? " ";:HY=H
Y-2:POSITION HX,HY:? CHR$(19)
420 IF J=13 AND HY<19 THEN POSITION HX,HY:? " ";:HY=
HY+2:POSITION HX,HY:? CHR$(19)
430 IF J=7 AND VX<17 THEN POSITION VX,VY:? " ";:VX=V
X+2:POSITION VX,VY:? CHR$(19)
440 IF J=11 AND VX>3 THEN POSITION VX,VY:? " ";:VX=V
X-2:POSITION VX,VY:? CHR$(19)
450 IF K=0 THEN GOSUB 1900
460 IF J<>15 THEN SOUND 0,20,10,8
470 FOR X=0 TO 10:NEXT X:SOUND 0,0,0,0:GOTO 300
1000 REM Raster fuellen
1010 PX=3:PY=2
1020 FOR Y=0 TO 14 STEP 2
1030 FOR X=0 TO 18 STEP 2
```

```
1040 POSITION PX+Y,PY+X:? CHR$(160)
1050 NEXT X:NEXT Y:POKE 764,255:RETURN
1100 REM Raster loeschen
1110 PX=3:PY=2
1120 FOR Y=0 TO 14 STEP 2
1130 FOR X=0 TO 18 STEP 2
1140 POSITION PX+Y,PY+X:? CHR$(32)
1150 NEXT X:NEXT Y:POKE 764,255:RETURN
1200 REM Invers ein und aus
1210 PX=3:PY=2
1220 FOR Y=0 TO 14 STEP 2
1230 FOR X=0 TO 18 STEP 2
1240 LOCATE PX+Y,PY+X,CH
1250 IF CH=32 THEN POSITION PX+Y,PY+X:? CHR$(160)
1260 IF CH=160 THEN POSITION PX+Y,PY+X:? CHR$(32)
1270 NEXT X:NEXT Y:POKE 764,255:RETURN
1300 REM Zeichen sichern
1310 FOR Y=0 TO 9:DW(Y)=0:NEXT Y
1315 POSITION 0,20:? "Player 0,1,2 oder 3 ";:INPUT Z
$
1320 POSITION 0,21:? "Ein Moment Geduld bitte";:CH=A
SC(Z$)
1325 DZ=ZN+CH:PX=3:PY=2
1330 FOR Y=0 TO 9:ZAE=0:FOR X=14 TO 0 STEP -2
1335 LOCATE PX+X,PY,CH:IF CH=160 THEN DW(Y)=DW(Y)+(2
^ZAE)
1340 ZAE=ZAE+1:NEXT X:PY=PY+2:NEXT Y
1345 POSITION 2,21:? CHR$(156);DZ;" DATA ";
1350 FOR Y=0 TO 8:? DW(Y);",";:NEXT Y:? DW(9)
1355 ? "CONT";:POSITION 2,19
1360 POKE 842,13
1365 END
1370 POKE 842,12
1375 POSITION 2,0:? CHR$(157)
1380 RESTORE :GOTO 20
1500 REM horizontal voll
1510 PX=3:PY=HY
1520 FOR X=0 TO 14 STEP 2:POSITION PX+X,PY:? CHR$(16
0):NEXT X
1530 POKE 764,255:RETURN
1600 REM horizontal leer
1610 PX=3:PY=HY
1620 FOR X=0 TO 14 STEP 2:POSITION PX+X,PY:? CHR$(32
):NEXT X
1630 POKE 764,255:RETURN
1700 REM vertikal voll
1710 PX=VX:PY=2
```

```
1720 FOR X=0 TO 18 STEP 2:POSITION PX,PY+X:? CHR$(16
0):NEXT X
1730 POKE 764,255:RETURN
1800 REM vertikal leer
1810 PX=VX:PY=2
1820 FOR X=0 TO 18 STEP 2:POSITION PX,PY+X:? CHR$(32
):NEXT X
1830 POKE 764,255:RETURN
1900 REM Punkt setzen und loeschen
1910 LOCATE VX,HY,CH
1920 IF CH=160 THEN POSITION VX,HY:? CHR$(32):SOUND
0,8,8,10
1930 IF CH=32 THEN POSITION VX,HY:? CHR$(160):SOUND
0,2,8,10
1940 RETURN
2000 REM Datei sichern
2010 POSITION 0,20:? "Eingabe Dateiname";:INPUT Z$
2020 POSITION 0,21:? "Ein Moment Geduld bitte";
2030 N$="D:":N$(3)=Z$
2040 LIST N$,10000,11000
2050 RESTORE :GOTO 20
6000 DATA 17,18,23,18,23,18,23,18,23,18,23,18,23,18,
23,18,5
6010 DATA 124,32,124,32,124,32,124,32,124,32,124,32,
124,32,124,32,124
6020 DATA 1,18,19,18,19,18,19,18,19,18,19,18,19,18,1
9,18,4
6030 DATA 124,32,124,32,124,32,124,32,124,32,124,32,
124,32,124,32,124
6040 DATA 26,18,24,18,24,18,24,18,24,18,24,18,24,18,
24,18,3
```

10: DW ist eine indizierte Variable für die Dezimalwerte der entworfenen Objekte. ZN ist die Anfangszeilennummer, ab der die Dezimalwerte später in DATA-Zeilen gespeichert werden.
30 bis 140: Hier wird das Entwurfsraster auf den Bildschirm gebracht. Es handelt sich dabei um die Grafikzeichen 17,18, 19,23 usw. (s. DATA-Zeilen 6000 bis 6040), die durch Drücken von CONTROL+Taste aufgerufen werden können
150: Die horizontalen und vertikalen Anfangspositionen für die Koordinatenkreuze, die vom Joystick gesteuert werden.
180 bis 290: Die Menüfunktionen erscheinen auf dem Bildschirm. Kursiv gelistete Buchstaben sollten Sie invertiert schreiben (ATARI-Taste drücken).
300: Die Abfrage nach einer gedrückten Taste (Register 764), nach Joystickbewegungen und Feuerknopf.
310 bis 380: In Register 764 wird der Tastaturcode für jede gedrückte Taste abgelegt. Dieser Code hat weder etwas mit ASCII noch mit dem internen Code zu tun (s. Tabelle im Anhang). Die Abfrage des Registers hat den großen Vorteil, daß der Rechner das Programm nicht unterbricht, wie es bei GET oder INPUT der Fall ist.
390 und 400: Die Speicherstelle 53279 registriert, wenn eine der Tasten OPTION, SELECT, START oder diese Tasten kombiniert miteinander gedrückt werden. OPTION bringt den Wert 3, SELECT den Wert 5.
410 bis 440: Bestimmung der Operationen, die je nach Joystickbewegung durchgeführt werden sollen. Wenn z.B. der Steuerknüppel nach vorn gedrückt wird, registriert STICK(0) eine 14. In diesem Fall soll das Koordinatenkreuz an der alten Position gelöscht werden und in der nächst höheren Reihe des Entwurfsrasters wieder erscheinen.
450: Wenn der Feuerknopf gedrückt wird, springt der Rechner ins Unterprogramm 1900.
460: Bei jeder Bewegung des Steuerknüppels entsteht ein Sound.
470: Eine sog. Verzögerungsschleife, die verhindert, daß die Koordinatenkreuze bei Joystickbewegungen zu schnell über den Bildschirm sausen.
1000 bis 1050: Das Unterprogramm zum Füllen des gesamten Entwurfrasters. Ein gesetzter Punkt erscheint als invertiertes Leerzeichen (ASCII-Code 160).
1100 bis 1150: Hier wird das gesamte Raster gelöscht (Leerzeichen mit ASCII-Code 32).
1200 bis 1270: Um ein Objekt zu invertieren, muß jeder Punkt der Matrix mit dem LOCATE-Befehl abgefragt werden (Variable CH). Wenn CH=32 ist, also ein Leerzeichen, dann soll an der betreffenden Stelle ein Punkt gesetzt werden; wenn sich hingegen ein gesetzter Punkt dort befindet (CH=160), so wird er gelöscht.

1300 bis 1380: Das Unterprogramm zur Sicherung der Dezimalwerte eines entworfenen Objektes. In Zeile 1310 werden zunächst alte Werte der indizierten Variable DW(x) gelöscht. Danach müssen Sie eingeben, um welchen Player es sich handelt. Die Variable DZ in Zeile 1325 bestimmt die DATA-Zeilennummer, in der die Dezimalwerte des entworfenen Objektes gespeichert werden. Die Grundzeilennummer ist 10000 (s. Zeile 10). Zu dieser Zeilennummer werden der ASCII-Code der eingegebenen Zahl (0,1,2 oder 3) gerechnet (z.B. bei 1=49). Die Daten werden dann in Zeile 10049 abgelegt. Der gesamte Vorgang des Speicherns geschieht automatisch. Durch die Zuordnung der DATA-Zeilen zu den Codezahlen erhalten Sie immer eine geordnete und übersichtliche Datei. Mit den Zeilen 1345 und 1350 werden die Zeilennummer, das Wort "DATA" und die einzelnen Dezimalwerte auf den Bildschirm gebracht. Mit einer automatischen RETURN-Funktion wird die DATA-Zeile in den Speicher aufgenommen. Der Vorgang geschieht so schnell, daß Sie ihn kaum verfolgen können. Sie erhalten anschließend ein neues Entwurfsraster und können den nächsten Player gestalten.

1500 bis 1830: In den Unterprogrammen 1000 bis 1150 wird das gesamte Entwurfsraster mit Punkten gefüllt bzw. gelöscht. Hier in den Zeilen 1500 bis 1830 werden diese Funktionen für einzelne Reihen horizontal bzw. vertikal durchgeführt.
1900 bis 1940: Um einen Zeichenpunkt (Bit) zu setzen oder zu löschen, muß beim anvisierten Matrixpunkt mit dem LOCATE-Befehl ermittelt werden, ob sich dort ein Leerzeichen oder ein Punkt befindet. Genau wie beim Invertieren wird dann ein Punkt gesetzt oder gelöscht.
2000 bis 2050: Da die DATA-Zeilen für die entworfenen Player bereits existieren, müssen zur Sicherung einer Datei nur die in Frage kommenden Zeilennummern im LIST-Format zur Diskettenstation übertragen werden. Wenn Sie einen Datenrecorder benutzen, ändern Sie die Variable N$ Zeile 2030 von "D:" in "C:".
6000 bis 6040: Die CHR$-Codes für das Entwurfsraster.

3.2 TESTBILD

Wie bereits erwähnt, können Sie mit Ihrem ATARI insgesamt 256 verschiedene Farbtöne darstellen. (Und wer die Wahl hat, hat die Qual.) Der Spruch ist zwar nicht mehr ganz frisch, aber zutreffend. Denn das Austüfteln, welche Farben in einem Programm zueinander passen, kostet viel Zeit. Ständig muß man die SETCOLOR- und COLOR-Werte ändern und von neuem starten.

Damit Sie künftig auf bequeme Weise Ihre Farbwahl für Player-Missile- und Hintergrundgrafik treffen können, finden Sie nachfolgend ein nützliches Utility. Hierbei erscheint auf dem Bildschirm ein Testbild, bei dem sich alle Farbfelder berühren, und eine Texteinblendung. Daneben erscheinen noch alle vier Player. Durch Tastendruck können Sie jedes einzelne Farbregister verändern und die Helligkeit bestimmen. Auf Wunsch werden die dazugehörenden SETCOLOR- und COLOR-Werte sowie die Dezimalwerte für der Farbregister der Player eingeblendet. Danach können Sie, falls Sie es wünschen, mit Ihren Kreationen fortfahren.

Bedienung von TESTBILD

1. Starten Sie mit RUN (RETURN). Durch Drücken der Tasten 1,2,3 und 4 ändern Sie die Farben der SETCOLOR-Register 0,1,2 und 4. Wenn Sie SHIFT + 1,2,3 oder 4 drücken, ändern Sie die Helligkeit der Farben. Durch Drücken von Q,W,E und R erhöhen sich die Dezimalwerte in den Farbregistern der Player.

2. Wenn Sie sich zwischendurch oder nach Abschluß Ihrer Farbwahl die betreffenden Werte ansehen wollen, drücken Sie die Leertaste. Um ins Testbild-Programm zurückzukehren, drücken Sie erneut die Leertaste.

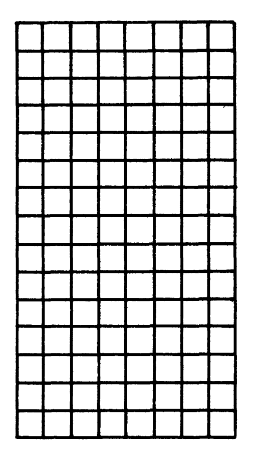
```
0 REM TESTBILD
10 POKE 106,PEEK(106)-4:GRAPHICS 3:? "MOMENT BITTE":
GOSUB 1000:? CHR$(125)
20 S0=0:S1=1:S2=2:S4=4:H0=6:H1=6:H2=2:H4=6
30 SETCOLOR 0,S0,H0:SETCOLOR 1,S1,H1:SETCOLOR 2,S2,H
2:SETCOLOR 4,S4,H4
40 COLOR 1:PLOT 19,9:DRAWTO 19,0:DRAWTO 0,0:POKE 765
,1:POSITION 0,9:XIO 18,#6,0,0,"S:"
50 COLOR 2:PLOT 39,9:DRAWTO 39,0:DRAWTO 20,0:POKE 76
5,2:POSITION 20,9:XIO 18,#6,0,0,"S:"
60 COLOR 3:PLOT 19,19:DRAWTO 19,10:DRAWTO 0,10:POKE
765,3:POSITION 0,19:XIO 18,#6,0,0,"S:"
```

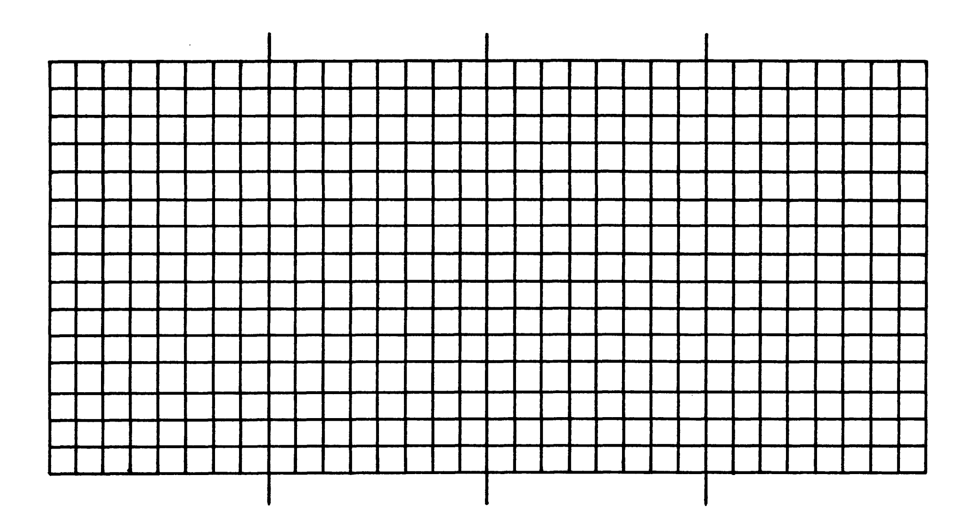
```
70 COLOR 0:PLOT 8,4:DRAWTO 8,9:PLOT 9,4:DRAWTO 9,9
80 COLOR 3:PLOT 28,4:DRAWTO 28,9:PLOT 29,4:DRAWTO 29
,9
90 COLOR 2:PLOT 8,10:DRAWTO 8,16:PLOT 9,10:DRAWTO 9,
16
100 COLOR 1:PLOT 28,10:DRAWTO 28,16:PLOT 29,10:DRAWT
O 29,16
110 POKE 752,1:CLOSE #1:OPEN #1,4,0,"K:"
120 ? :? " T E X T   U N D   T E S T B I L D"
130 GET #1,F
140 IF F=49 THEN S0=S0+1:IF S0>15 THEN S0=0
150 IF F=50 THEN S1=S1+1:IF S1>15 THEN S1=0
160 IF F=51 THEN S2=S2+1:IF S2>15 THEN S2=0
170 IF F=52 THEN S4=S4+1:IF S4>15 THEN S4=0
180 IF F=33 THEN H0=H0+2:IF H0>14 THEN H0=0
190 IF F=34 THEN H1=H1+2:IF H1>14 THEN H1=0
200 IF F=35 THEN H2=H2+2:IF H2>14 THEN H2=0
210 IF F=36 THEN H4=H4+2:IF H4>14 THEN H4=0
211 IF F=81 THEN FP0=FP0+2:IF FP0>254 THEN FP0=0
212 IF F=87 THEN FP1=FP1+2:IF FP1>254 THEN FP1=0
213 IF F=69 THEN FP2=FP2+2:IF FP2>254 THEN FP2=0
214 IF F=82 THEN FP3=FP3+2:IF FP3>254 THEN FP3=0
220 IF F=32 THEN GOTO 300
230 SETCOLOR 0,S0,H0:SETCOLOR 1,S1,H1:SETCOLOR 2,S2,
H2:SETCOLOR 4,S4,H4
240 POKE 704,FP0:POKE 705,FP1:POKE 706,FP2:POKE 707,
FP3
250 GOTO 130
300 GRAPHICS 0
310 POKE 710,194:POKE 752,1:POKE 53277,0
320 POSITION 4,4:? "Farbwerte:"
330 POSITION 4,7:? "SETCOLOR 0,";S0;",";H0;"  (COLOR
 1)"
340 POSITION 4,9:? "SETCOLOR 1,";S1;",";H1;"  (COLOR
 2)"
350 POSITION 4,11:? "SETCOLOR 2,";S2;",";H2;"  (COLO
R 3)"
360 POSITION 4,13:? "SETCOLOR 4,";S4;",";H4;"  (COLO
R 0)"
361 POSITION 4,15:? "Player 0, Register 704";",";FP0
362 POSITION 4,17:? "Player 1, Register 705";",";FP1
363 POSITION 4,19:? "Player 2, Register 706";",";FP2
364 POSITION 4,21:? "Player 3, Register 707";",";FP3
370 GET #1,F
380 IF F=32 THEN GRAPHICS 3:RESTORE :GOSUB 1050:GOTO
 30
390 GOTO 370
```

```
1000 POKE 54279,PEEK(106)
1010 PMT=PEEK(106)*256:HP=50:VP=90:FP0=2:FP1=2:FP2=2
:FP3=2
1020 FOR L=PMT+384 TO PMT+1024
1030 POKE L,0:NEXT L
1040 P0=PMT+512:P1=P0+128:P2=P1+128:P3=P2+128
1050 FOR L=0 TO 9:READ D:POKE P0+L+VP,D:NEXT L
1060 FOR L=0 TO 9:READ D:POKE P1+L+VP,D:NEXT L
1070 FOR L=0 TO 9:READ D:POKE P2+L+VP,D:NEXT L
1080 FOR L=0 TO 9:READ D:POKE P3+L+VP,D:NEXT L
1090 POKE 559,42:POKE 623,1
1100 POKE 53277,2
1110 POKE 704,FP0:POKE 705,FP1:POKE 706,FP2:POKE 707
,FP3
1120 POKE 53248,HP:POKE 53249,HP+40
1130 POKE 53250,HP+80:POKE 53251,HP+120
1140 RETURN
1150 DATA 255,255,255,255,255,255,255,255,255,255,25
5
1160 DATA 255,255,255,255,255,255,255,255,255,255,25
5
1170 DATA 255,255,255,255,255,255,255,255,255,255,25
5
1180 DATA 255,255,255,255,255,255,255,255,255,255,25
5
```

10: Für zweizeilige Player-Missile-Grafik werden vom RAMTOP aus 4 "Seiten" reserviert. Das Testbildes wird in GRAPHICS 3 dargestellt.
20: Die Anfangswerte der einzelnen Farbregister und Helligkeitsstufen.
40 bis 100: Das Testbild wird auf den Bildschirm gebracht. Dabei berühren sich alle Farbfelder. Der XIO-Befehl bewirkt das Ausfüllen einer Fläche. Dieses praktische Kommando erspart das zeilenweise PLOTten, das normalerweise zum Ausfüllen einer Fläche notwendig wäre.
110: 752,1 macht den Cursor unsichtbar. Der OPEN-Befehl öffnet den Datenkanal zur Tastatur.
130 bis 220: Je nach gedrückter Taste (1,2,3,4 und SHIFT + 1, 2,3,4 oder Q,W,E,R) wird der Wert für das betreffende Farbregister oder die Helligkeitsstufe erhöht.
230: Die aktuellen Farbwerte werden auf dem Bildschirm dargestellt.
300: In dieses Unterprogramm springt der Rechner, wenn Sie die Leertaste drücken. Sie können sich dadurch die aktuellen Farbwerte auflisten lassen.

370 bis 390: Durch erneutes Drücken der Leertaste gelangen Sie wieder zum Testbild.
1000 bis 1180: Aufbau der Player-Missile-Tabelle, so wie im ersten Kapitel des Buches ausführlich beschrieben.





Player-Missile Entwurfsraster (zum Fotokopieren)

TASTATUR-CODE							
	gedrückte Taste(n)				gedrückte Taste(n)		
Code		+64	+128	Code		+64	+128
0	L	l	▪	32	,	[	♥
1	J	j	◣	33	leer		
2	;	:	♣	34	.	]	♦
3				35	N	n	—
4				36			
5	K	k	▪	37	M	m	—
6	+	\	←	38	/	?	
7	*	^	→	39	inv		
8	O	o	▪	40	R	r	—
9				41			
10	P	p	♣	42	E	e	┐
11	U	u	▬	43	Y	y	▌
12	Ret			44	▸	▸	▸
13	I	i	▪	45	T	t	●
14	-	_	↑	46	W	w	┬
15	=	|	↓	47	Q	q	┌
16	V	v	|	48	9	(	
17				49			
18	C	c	┘	50	0	)	
19				51	7	'	
20				52	BackS		Del
21	B	b	|	53	8	@	
22	X	x	┴	54	<	Clear	
23	Z	z	└	55	>	Ins	
24	4	$		56	F	f	╱
25				57	H	h	◢
26	3	#		58	D	d	┤
27	6	&		59			
28	ESC			60	Caps		
29	5	%		61	G	g	╲
30	2	"	BEL	62	S	s	┼
31	1	!		63	A	a	├

In der Adresse 764 wird die zuletzt gedrückte Taste im Tastatur-Code erfaßt. Der Tastatur-Code ist eine 6-Bit-Information (Dezimalwerte von 0 bis 63). Wird zusammen mit einer Taste SHIFT gedrückt, wird Bit 6 (dezimal 64) gesetzt. Das Drücken von CONTROL setzt Bit 7 (dezimal 128).

TASTATUR-Code

♥	160	SPACE	33	@	117	♦	162
├	191	!	95	A	63	a	127
¦	149	"	94	B	21	b	85
┘	146	#	90	C	18	c	82
┤	186	$	88	D	58	d	122
┐	170	%	93	E	42	e	106
╱	184	&	91	F	56	f	120
╲	189	'	115	G	61	g	125
◢	185	(	112	H	57	h	121
▗	141	)	114	I	13	i	77
◣	129	*	7	J	1	j	65
▝	133	+	6	K	5	k	69
▘	128	,	32	L	0	l	64
▔	165	-	14	M	37	m	101
▁	163	.	34	N	35	n	99
▖	136	/	38	O	8	o	72
♣	138	0	50	P	10	p	74
┌	175	1	31	Q	47	q	111
─	168	2	30	R	40	r	104
┼	190	3	26	S	62	s	126
●	173	4	24	T	45	t	109
▄	139	5	29	U	11	u	75
▎	144	6	27	V	16	v	80
┬	174	7	51	W	46	w	110
┴	150	8	53	X	22	x	86
▌	171	9	48	Y	43	y	107
└	151	:	66	Z	23	z	87
␛	28	;	2	[	96	♠	130
↑	142	<	54	\	70			79
↓	143	=	15	]	98	CLEAR	118
←	134	>	55	^	71	BACK	52
→	135	?	102	_	78	TAB	44

39	ATARI	119	INSERT
60	CAPS	118	CLEAR
167	CONTROL+ATARI	172	CONTROL+TAB
12	RETURN	108	SHIFT+TAB
116	DELETE	158	CONTROL+2 (BEL)

ZEICHENSATZ

ATASCII-Code – Zeichen – interner Code

A	Z	I	A	Z	I	A	Z	I
0	♥	64	40	(	8	80	P	48
1	├	65	41	)	9	81	Q	49
2	▕	66	42	*	10	82	R	50
3	┘	67	43	+	11	83	S	51
4	┤	68	44	,	12	84	T	52
5	┐	69	45	-	13	85	U	53
6	╱	70	46	.	14	86	V	54
7	╲	71	47	/	15	87	W	55
8	◢	72	48	0	16	88	X	56
9	▗	73	49	1	17	89	Y	57
10	◣	74	50	2	18	90	Z	58
11	▝	75	51	3	19	91	[	59
12	▘	76	52	4	20	92	\	60
13	▔	77	53	5	21	93	]	61
14	▁	78	54	6	22	94	^	62
15	▖	79	55	7	23	95	_	63
16	♣	80	56	8	24	96	♦	96
17	┌	81	57	9	25	97	a	97
18	─	82	58	:	26	98	b	98
19	┼	83	59	;	27	99	c	99
20	●	84	60	<	28	100	d	100
21	▄	85	61	=	29	101	e	101
22	▎	86	62	>	30	102	f	102
23	┬	87	63	?	31	103	g	103
24	┴	88	64	@	32	104	h	104
25	▌	89	65	A	33	105	i	105
26	└	90	66	B	34	106	j	106
27	␛	91	67	C	35	107	k	107
28	↑	92	68	D	36	108	l	108
29	↓	93	69	E	37	109	m	109
30	←	94	70	F	38	110	n	110
31	→	95	71	G	39	111	o	111
32		0	72	H	40	112	p	112
33	!	1	73	I	41	113	q	113
34	"	2	74	J	42	114	r	114
35	#	3	75	K	43	115	s	115
36	$	4	76	L	44	116	t	116
37	%	5	77	M	45	117	u	117
38	&	6	78	N	46	118	v	118
39	'	7	79	O	47	119	w	119

Code	Char	Code	Code	Char	Code	Code	Char	Code
120	x	120	166		134	212		180
121	y	121	167		135	213		181
122	z	122	168		136	214		182
123		123	169		137	215		183
124	|	124	170		138	216		184
125		125	171		139	217		185
126		126	172		140	218		186
127		127	173		141	219		187
128		192	174		142	220		188
129		193	175		143	221		189
130		194	176		144	222		190
131		195	177		145	223		191
132		196	178		146	224		224
133		197	179		147	225		225
134		198	180		148	226		226
135		199	181		149	227		227
136		200	182		150	228		228
137		201	183		151	229		229
138		202	184		152	230		230
139		203	185		153	231		231
140		204	186		154	232		232
141		205	187		155	233		233
142		206	188		156	234		234
143		207	189		157	235		235
144		208	190		158	236		236
145		209	191		159	237		237
146		210	192		160	238		238
147		211	193		161	239		239
148		212	194		162	240		240
149		213	195		163	241		241
150		214	196		164	242		242
151		215	197		165	243		243
152		216	198		166	244		244
153		217	199		167	245		245
154		218	200		168	246		246
155		219	201		169	247		247
156		220	202		170	248		248
157		221	203		171	249		249
158		222	204		172	250		250
159		223	205		173	251		251
160		128	206		174	252		252
161		129	207		175	253		253
162		130	208		176	254		254
163		131	209		177	255		255
164		132	210		178			
165		133	211		179			